EFFICIENT MODELING OF HIGHER ORDER AND LONGER RANGE GEOMETRY STATISTICS

A Dissertation

Presented to the Faculty of the Graduate School

of Cornell University

in Partial Fulfillment of the Requirements for the Degree of

Doctor of Philosophy

by

Yimeng Zhang

January 2013

EFFICIENT MODELING OF HIGHER ORDER AND LONGER RANGE GEOMETRY STATISTICS

Yimeng Zhang, Ph.D.

Cornell University 2013

The local feature based approaches have become popular in most vision applications. A local feature captures the local appearance of objects or scenes, and is more robust to environment and view-point changes comparing to features extracted from the entire image. The shape and context information is further captured with the spatial relationships of the local features. Modeling more spatial information usually leads to exponential or polynomial increase of the computational cost. Therefore, the spatial modeling of prior work is limited to neighboring or weak geometry relationships of local features, or is not view-point invariant.

In this thesis, we propose algorithms that model *rich* geometry information with *little* sacrifice of the computational cost. We focus on two main vision problems, the whole image representation and the pixel-level image labeling. For each of them, we present an algorithm that incorporate spatial information to its most popular and basic technique: the Bag-of-Words (BoW) representation and Conditional Random Field (CRF) model respectively. Our proposed algorithm is general enough to be applied to or combined with any other advanced technique, which utilizes BoW or CRF as part of it, to further improve its performance with only little increase of the computational cost. We show example usages of the proposed algorithms in several applications, including object recognition, object localization, image retrieval, activity recognition in

videos, and object-based image segmentation. Experiment results show that our approaches improve the performance of the state-of-arts for these applications with only little increase of the computation cost.

BIOGRAPHICAL SKETCH

Yimeng Zhang finished her Ph.D. in the School of Electrical and Computer Engineering in Cornell University in 2012. Her Ph.D. research is on computer vision and machine learning. Before coming to Cornell, she received her Master degree in the School of Computer Science at Carnegie Mellon University in 2008. Before coming to the US, she did her undergraduate study in Kyoto University, Japan and graduated in 2006. She was originally born and grown in Shenyang, China.

This thesis is dedicated to my parents and my grandparents.

ACKNOWLEDGEMENTS

These years of my Ph.D. life have been full of excitement and extremely enjoyable. I would like to thank everyone, who has made this happen.

First, I would like to thank my advisor, Prof. Tsuhan Chen. I have been working with Prof. Chen since I started my master study six years ago, when I did not even know what "research" means or how to start a "project". He has given me the best guidance and perspective at every phrase of my research. More importantly, he has put a lot of efforts to make me a good researcher on my own. At the same time, he has provided me so much freedom for working on things I am interested in and has created an environment where I can truly enjoy my research.

My great gratitude also goes to my husband, Tian Gao, who is always the first one I would think of relying on whenever anything bad happens and always the first one I am eager to share with for any good news in my life. He has given me the largest support on everything and always tried his best to make me happy. The same thank also gives to my parents, Qiang Zhang and Linghua Li. They have been the main origin of my confidence and success because they made me believe that I can always seek for their unconditional support and love whenever I need it.

I also thank my thesis committees, Prof. Thorsten Joachims, Prof. Xiaoming Liu, and Prof. Anthony Reeves. Thank you for many invaluable suggestions and insightful ideas on my research and thank you for your kindness and patience to help me go through those challenging exams for finishing my Ph.D.

Last but not least, I would like to thank all the AMP lab members and alumni who have overlaps with me. Many research ideas can not be generated and many work cannot be improved so well without those discussions in the group

meeting or in private. Moreover, the life at school would not be so happy without these nicest people.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

In most vision techniques, images and videos are represented with local components, which capture the local appearances, such as edges, texture, and color, or local movements. The global shapes or contexts are modeled using the spatial information of the local components. Modeling the spatial information is challenging. On the one hand, we must encode enough spatial information in order to achieve enough precision of the vision tasks. For example, object recognition requires enough shape information to discriminate different object categories. On the other hand, the shape information we encoded should be flexible enough to overcome the large intra-category variations, such as location or scale changes. Moreover, the resulting system should be computationally efficient enough.

This thesis is focusing on efficient algorithms that model rich spatial information of the local components. Existing methods on this problem usually suffer one or more of the following problems: 1) they extensively increase the computational cost by including more spatial information; 2) the spatial information is variant to pose changes, therefore, the methods require the objects in the images aligned or require evaluating sub-windows in images; 3) the methods only encode weak spatial information, such as the neighborhood of each local component, or only distances between pairs of local components. In this thesis, we propose a family of algorithms that model higher order and longer range spatial information among the local components with little sacrifice of the invariance and computational cost.

We have explored two main vision problems: 1) image representation and 2) pixel-level image labeling. The main goal of the first problem is to obtain a numerical representation of the whole image so as to perform the tasks like object recognition/detection and image retrieval. Applications for pixel-level image labeling, such as image segmentation, tries to predict a label for every pixel instead of the whole image. For both problems, we have proposed algorithms that incorporate richer spatial information with much more efficiency comparing to prior work.

We propose our main algorithm for encoding spatial information to the whole image representation in Chapter 2, and its three variations for three applications: object recognition in Chapter 3, object detection in Chapter 4 and image retrieval in Chapter 5. In Chapter 6, we extend this algorithm to the representation of videos by adding a temporal dimension. Then, we present the algorithm that incorporates spatial information for the pixel-level image labeling problem and its application in Chapter 7. Finally, we conclude our work and point to future work in Chapter 8.

CHAPTER 2

IMAGE REPRESENTATION: HIGH ORDER FEATURES

2.1 Introduction

Most recent vision techniques represent an image with local features. In terms of the global geometry information, the Bag of words (BoW) representation [17] takes one extreme. It discards all spatial information of the local features. Due to its computational efficiency and intra-category invariance, BoW has been the most popular representation for many vision applications, such as object recognition, object detection and image retrieval. However, since it lacks the spatial modeling, the discrimination power is limited.

To incorporate spatial information to the BoW representation, one type of technique uses mutual geometry relationship between the local words [63, 74, 96, 118, 129, 133, 140]. Higher Order Features (HOFs) are constructed with a specific number of local words, together with their spatial layout. Following the definition from [74], we call the local features $1st$ order features, and features with two, three, n words, $2nd$, $3rd$ and n^{th} order features. The main advantage of this type of methods is the invariance to translation or scale changes. However, as the order increases, the dimension of the feature vector will increase exponentially to the order, and immediately reach an intractable amount. For example, when the vocabulary size is 4000 and the image space is modeled with a 20×20 grid, the dimension of 2nd order feature vector would be more than 1 billion, and as large as 10^{19} for the 3rd order. To reduce the feature dimension, previous works only use up to 3rd order features [74, 118, 133, 140]. Moreover, the HOFs are usually created with local words in a short range with weak ge-

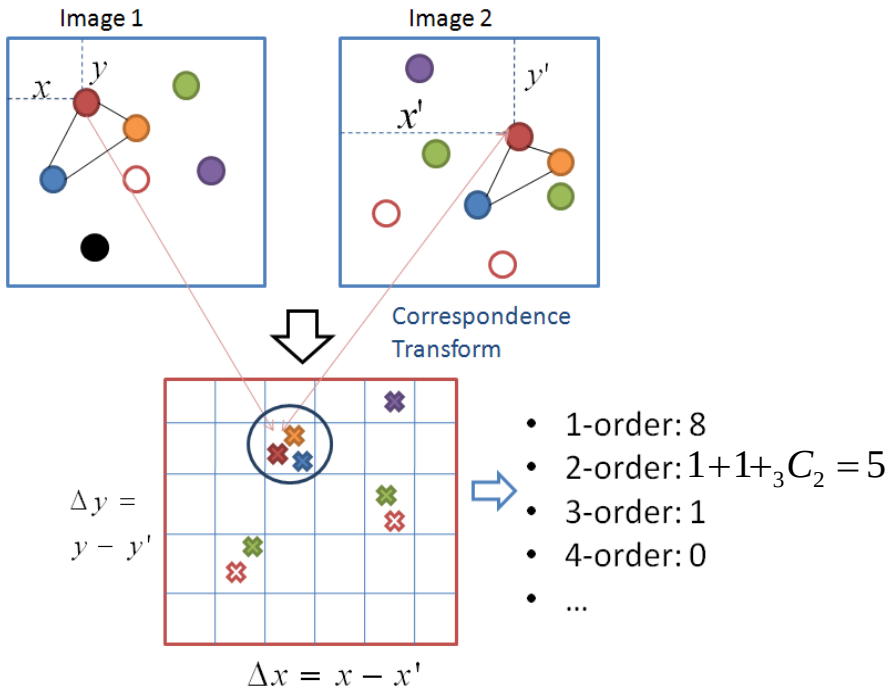


Figure 2.1: Illustration of the basic idea. Each circle in the top two images corresponds to a visual word (local feature). Different colors represent different words. Two images are transformed to the offset space (bottom image) in order to find the co-occurrence of high order features. Each cross in the offset space is created by a pair of same words (same color) form the two input images. The main idea is that when n points have the same location in the offset space, we have a particular co-occurring n order feature.

ometry, such as the co-existences in the same image or neighboring information [63, 96, 111, 129, 133, 140], and thus ignore long range interactions.

We propose an efficient algorithm which is capable of handling unbounded (one to *infinite*) order features with rich geometry. The main idea is illustrated in Figure 2.1. In order to find the co-occurring HOFs between two images, we transform the local features to the offset space. A vote is created in the offset space for each same word pair in the two images. The location in the offset space is the relative location of the two words. After transforming to the offset

space, n votes at the same location indicate n words of the same mutual spatial layout in the two images, which correspond to a co-occurring n^{th} order feature. After the transformation, it would be quite easy to find the co-occurring HOFs, which is intractable in the original image space. Figure 7.1 illustrates the benefit of matching two images using the HOFs detected with our algorithm. With the visual words, we have many matches for both images of different objects (left) and images of the same object (right). On the contrary, with the HOFs, we have much fewer matches for different objects than the same object. When we increase the order to 3, we cannot find any matches for the two irrelevant images.

We can further prove that the number of co-occurring n^{th} order features equals to the inner product of the histograms of n^{th} order features of the two images. In standard procedure for computing the inner product, the system first computes the histograms, and then the distance between the histograms. Both of these steps require exponentially large computation. In comparison, our algorithm computes one to infinite order features with the same computational complexity as BoW.

For different vision applications, we can easily integrate the efficiently computed inner-product of HOF histograms to other state-of-arts methods to replace original BoW representation. In object recognition (Chapter 3) and object detection (Chapter 4), we use the inner product as a kernel for the SVM. For object detection (Chapter 4), we also propose an efficient sub-window search algorithm for the HOFs to avoid evaluating every sub-windows in an image. For the large-scale image retrieval application (Chapter 5), we use the proposed algorithm to calculate the cosine similarity of the high order feature vectors of

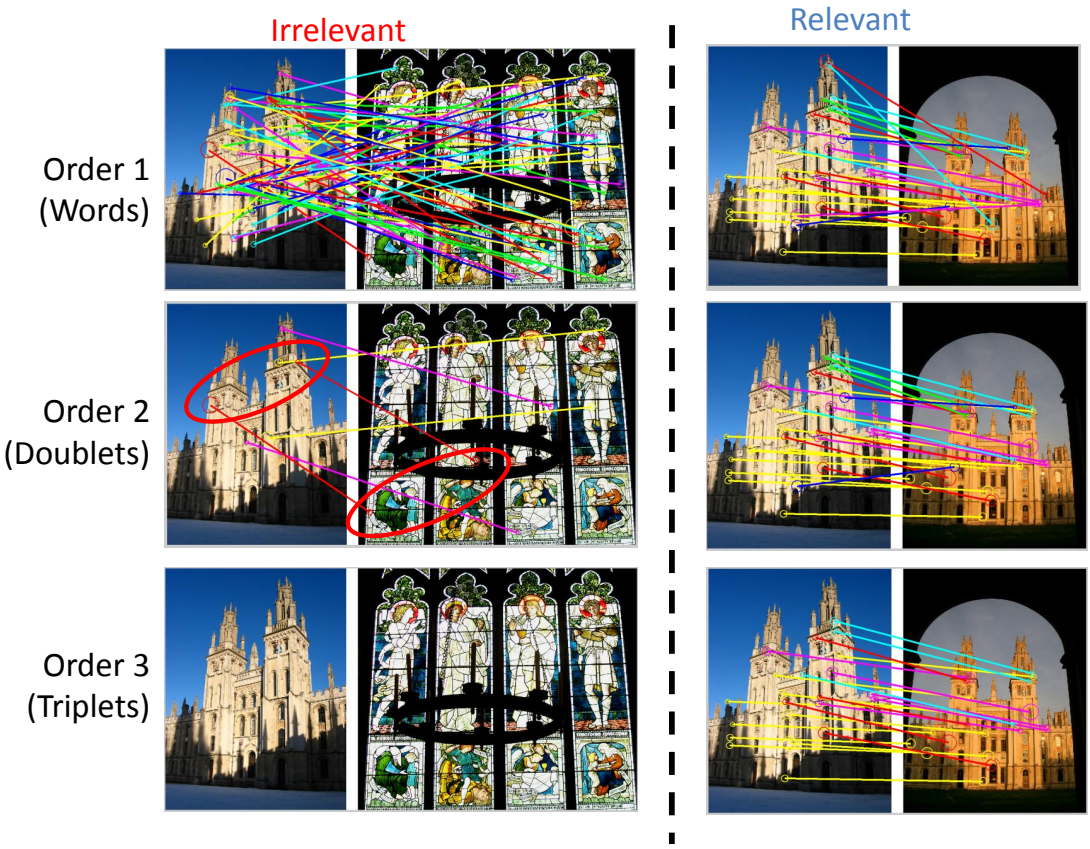


Figure 2.2: The images on the left show the matched *High-Order Features (HOF)* of different orders for two irrelevant images, while the images on the right show the matches for two relevant images (images of the same object). The top row shows the matched *visual words*, which also correspond to the 1st order features, and the second and third rows show the 2nd order (doublets), and the 3rd order (triplets) features respectively. With the visual words, we have many matches for both images of different objects (left) and images of the same object (right). On the contrary, with the HOFs, we have much fewer matches for different objects than the same object.

the query image and images in the database. To avoid comparing the query with every image in the database, we integrate the proposed algorithm with the inverted files structure, and remain the computational cost similar to that of the BoW method for image retrieval. For activity recognition from videos

(Chapter 6), we further extend the algorithm from 2D to 3D by adding the temporal domain to capture both the spatial and temporal relationships of the local movements.

2.2 Related Work

2.2.1 BoW and Beyond

One of the most popularly used technique in computer vision is the Bag-of-Words (BoW) representation of images [17, 83, 88, 106, 137]. Local features are detected at interest regions or sampled densely from the images, and then are assigned to visual words that are learned usually with K-means. The BoW discards any geometry information of the local features and represents an image as a histogram of the visual words. Due to its computational simplicity and robustness to transformation variances, BoW has gain great success in many applications, such as object recognition, detection and image retrieval.

Recent work that improve BoW are mainly on: 1) encoding methods that create a better word representation for each local feature, such as sparse coding [130] or locality sensitive coding [120]; 2) pooling techniques that aggregate the words/codes in an image to a vector representation, such as average pooling (same as histogram) and max pooling [120]. Geometric l_p-norm pooling [28] and receptive fields pooling [45] learns the features that are most important for classification to select from images; 3) approaches that encode geometry and shape information to BoW by modeling the locations of the words. This paper is along the third line of work. Our approach is general enough to be easily

combined with advances from the first two types of work.

2.2.2 Spatial Modeling

We will review the prior works that incorporate the spatial information into the BoW representation in details for different applications in Chapter 3, 4, 5, and 6. Generally speaking, there are mainly four types of methods: 1) Spatial Pyramid Matching (SPM) [21, 65, 125, 130] based methods incorporate rigid spatial information by quantizing the image space into grid regions and thus are not translation or scale invariant; 2) star shaped models [25,66,81,132], which models the location distribution of the words relative to the center of the object, and thus require object bounding boxes for the training images. 3) Constellation model [29] constructs a graph with latent nodes representing a fixed number of parts, and is translation and scale invariant. However, this type of models is usually computational expensive; 4) High Order Feature based models [63, 74, 96, 102, 118] construct composite features that capture the mutual geometrical relationship between the local words. Our work belongs to this type. Comparing to prior work, our algorithm is more efficient so as to compute much higher (up to infinite) order features and encode much richer geometry.

2.3 Main Algorithm

2.3.1 Image Representation

An image I is represented as a collection of local patches, $I = \{p_1, ..., p_m\}$. Each patch is represented with its visual word assignment w, region size s and location x, y. $p_i = (w_i, s_i, x_i, y_i)$.

2.3.2 High-Order Feature

We define the high order features (HOF) with one word 1st order features, and features with two, three, n words, 2nd, 3rd and n^{th} order features. Different relative spatial distribution among the n visual words yields different n^{th} order features. Figure 2.1 shows example occurrences of the same 3rd order feature in two images. An image will be represented as the histogram defined with the HOFs. The dimension of this histogram can be extremely long even when $n = 2$ while a large vocabulary is used. Therefore, it is impractical to create the histograms directly and perform the computation.

2.3.3 Correspondence Transform

For the explanation simplicity, we first only consider the translation invariance for the HOF. The main idea of the algorithm is that if two HOF co-occur in two images, they must be a translation from one image to the other. Thus, we can simplify the task of counting co-occurrence of n^{th} order features in two images

to counting the constant shift n visual word pairs in the two images. However doing this directly on the image space would be still too much. In order to facilitate this process, we first transform the feature points in two images to the offset space. The idea is illustrated in Figure 2.1. We call this "Correspondence Transform". For each pair of same words in the two images, we calculate their offset $(\triangle x, \triangle y)$, which is the location of the word in image I_1 minus that in image I_2. Then a vote is generated in the offset space at $(\triangle x, \triangle y)$. If there are multiple correspondences for the same word, we create multiple votes, as the green and red words. In the offset space, n votes locating at the same position corresponds to a co-occurring n^{th} order feature. Thus, to count the number of co-occurring n^{th} order features in the two images, we can simply count the number of n votes at the same location in the offset space. In the example of figure 2.1, the number of co-occurring 2nd order features is $1 + 1 + \binom{3}{2}$; while the number for 3rd order feature is 1, since we only have 1 position with 3 votes. And for all $n > 3$, the number is 0 since we do not have larger than 3 votes at the same location. We summarize the algorithm as in Algorithm 1.

Algorithm 1: Compute the number of co-occurring HOFs
Input: Two images represented with visual words: $I_1 = \{v_i\}, I_2 = \{v_j\}$
Output: Number of co-occurring n^{th} order features, $K_n(I_1, I_2)$, for all $n = 1, 2, ..., \infty$
Algorithm:
1. Perform correspondence-transform:
(a) For each pair of same words in the two images ($v_i \in I_1, v_j \in I_2$) , compute their offset ($\Delta x = x_i - x_j, \Delta y = y_i - y_j$).
(b) Create a vote in the offset space at $(\Delta x, \Delta y)$
2. In the quantized offset space, for each bin that has $N (N \geq n)$ votes:
Increment $K_n(I_1, I_2)$ with N choose n, $\binom{N}{n}$.
$K_n(I_1, I_2) = K_n(I_1, I_2) + \binom{N}{n}$.

2.3.4 Relationship with HOF Histogram

We analyze the relationship between the co-occurring HOFs of two images and their HOF histograms. The n^{th} order histogram of an image I is a vector $\Phi(I)$ with the f_n coordinate as the number of occurrences of feature f_n. The inner-product of the HOF histograms of two images can be computed as follows.

$$< \Phi(I_1), \Phi(I_2) > \tag{2.1}$$

$$= \sum_{f_n} < \Phi_{f_n}(I_1), \Phi_{f_n}(I_2) > \tag{2.2}$$

$$= \sum_{f_n} \sum_{u_n = f_n, u_n \in I_1} 1 \times \sum_{u_n = f_n, u_n \in I_2} 1 \tag{2.3}$$

$$= \sum_{f_n} \sum_{u_n = f_n, u_n \in I_1} \sum_{u_n = f_n, u_n \in I_2} 1. \tag{2.4}$$

where $u_n = f_n, u_n \in I_1$ indicates that u_n is an occurrence of feature f_n in image I_1.

Therefore, the inner product of the n^{th} order histograms of two images equals

to the total number of co-occurring n^{th} order features. We example the usage of this inner product for different applications in the later sections. Notice that if we directly compute the histograms and then compute the inner-product, the computation time would be $|\Sigma|^n \times |X|^{n-1}|Y|^{n-1}$, where $|\Sigma|$ is the visual word vocabulary size and X, Y are the X and Y axes of the image space. While using the proposed algorithm, the running time is the linear to the number of same word pairs. Moreover, with this computation, we have calculated all order (one to *infinite*) features.

2.3.5 More Invariance

Adding more invariance is as easy as adding more dimension to the offset space. Taking scale invariance as example, as illustrated in figure 2.3, we add a scale dimension to the offset space. Let (x_i, y_i) denote the position of word i, and s_i denote the size of the region used to create this word. For each pair of same words i and i' in the two images, we create a vote at $(\hat{x}, \hat{y}, \hat{s}) = (x_i - \frac{s_i}{s'_i}x'_i, y_i - \frac{s_i}{s'_i}y'_i, log(\frac{s_i}{s'_i}))$. In the 3D offset space, if we have n votes at the same location, we have a co-occurring n^{th} order feature with both translation and scale invariance. In figure 2.3, at scale difference $\hat{s} = 1/2$, we have a co-occurring 3rd order feature. Rotation invariance can be similarly incorporated.

2.3.6 Computation Time

Let P denote the number of same word pairs of two images. It takes $O(P)$ time to calculate the offset space with Algorithm 1. In the worst case (all features in

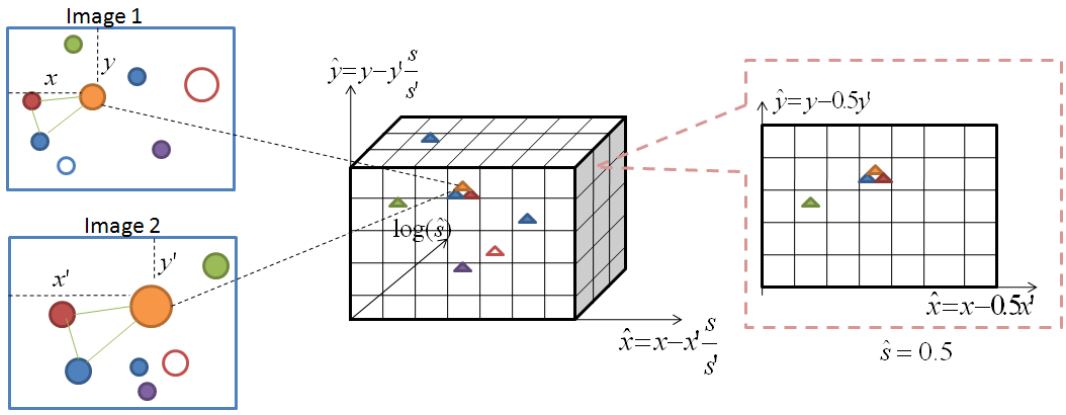


Figure 2.3: Illustration of the algorithm for finding co-occurrences of scale-invariant high order features. Each circle in the images is a local patch. The different colors represents different visual word assignments. The size of the circle represents the scale of the patch. A triangle corresponds to a vote generated by a pair of patches.

the two images are assigned to the same word), we have $P = O(M^2)$ same word pairs, where M is the number of features per image. However, in practice, the number of same word pairs is linear to M. Especially with a large vocabulary size, P is usually even smaller than M. Therefore, in practice we only need $O(M)$ time to compute all (1 to *infinite*) order features, which is the same complexity as BoW.

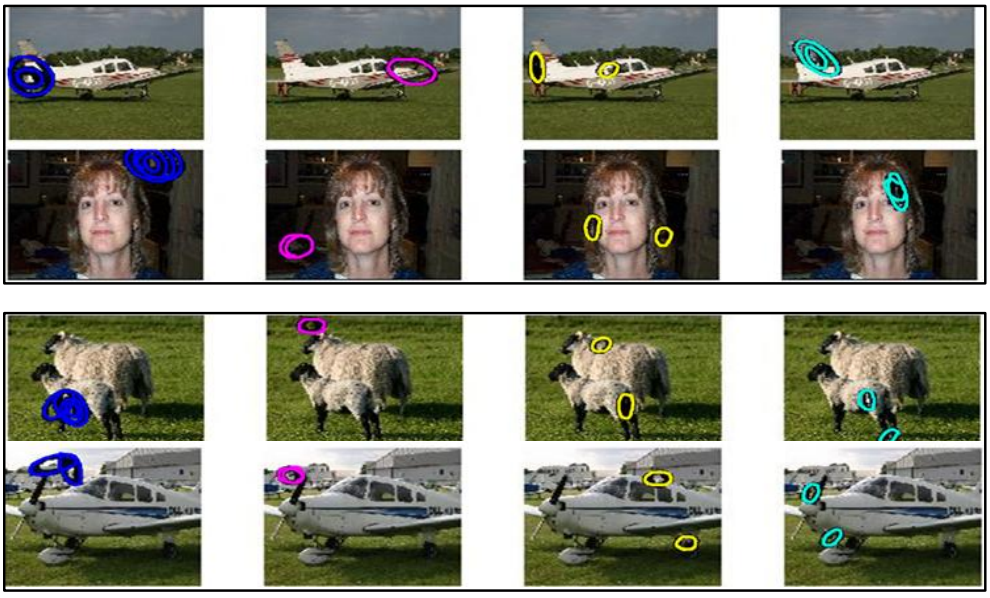
2.4 Example Visualization

Figure 2.4 shows the example co-occurring HOF in two images detected with our algorithm. Figure 2.5 shows the example scale-invariant HOF detected in two images.

2.5 Different Applications

For different vision applications, we make different usage of the inner product of HOF histograms computed efficiently with the proposed algorithm. In the following chapters, we will present four example usages: object recognition, object detection, image retrieval, and activity recognition from videos.

(*a*)



(*b*)

Figure 2.4: Example co-occurring HOFs. For each pair of images, we show the HOF of the four largest order features being detected (from left to right). Different color represents different co-occurring features. At the top (a), the two images are from the same category. At the bottom (b), images are from different categories. We can find larger order features for images from the same category.

Figure 2.5: Example scale invariant HOF detected with our algorithm. For each pair of images, we show the co-occurring HOF of the largest order detected.

CHAPTER 3

HOF FOR OBJECT RECOGNITION

The goal of the object recognition task is to determine whether an object is included in an image. This is essentially a classification task, which classifies an image into a pre-defined object category. Bag-of-Words (BoW) model has gain great success in this task [17,137].

3.1 Related Work

For the task of object recognition, the most widely used method for encoding geometry information to BoW is the Spatial Pyramid Matching (SPM) [11,65]. SPM incorporates rigid spatial information by quantizing the image space into grid regions, and has achieved the state-of-the-art performance when combined with advanced coding and pooling techniques [28,45,120], on datasets like Caltech-101 [93] where objects are centered and aligned in images. However it lacks invariance to translation or scale differences of the objects. Another type is the star shaped models [81], which model the location distribution of the words relative to the center of the object, and thus require object bounding boxes for the training images. Constellation model [29] represents the objects with a fixed number of parts and captures the spatial layout of the parts with a joint Gaussian, and can be translation and scale invariant. However, this type of models is computationally expensive in that it requires searching an exponentially large number of hypothesis which give different part assignments to the features. Graph based models [22,67] find a geometrically consistent matching between features from two images by optimizing a graph, which has a node for each pair of local fea-

tures, and thus can be quite large.

Another extension for improving the BoW representation is to model the mutual geometry relationship among the words [63, 74, 96, 118]. Our work belongs to this type. Comparing to prior work, our algorithm is more efficient so as to compute much higher (up to infinite) order features and encode richer geometry.

3.2 HOF as SVM Kernel

3.2.1 HOF Kernel

We use the inner-product of HOF histograms as the kernel for the Support Vector Machine (SVM), which is equivalent to performing classification with SVM in the HOF space rather than the word space. To define a metric as a SVM kernel, it is known that the Mercer's condition (positive semi-definite) must be satisfied so as to guarantee that the learning of the SVM results in a global optimal. Since we define the kernel as an inner product, it satisfies the Mercers condition from its definition.

The co-occurrence can be efficiently computed with the algorithm proposed in previous Section. Specifically, the kernel $K_n(I_1, I_2)$ of the n^{th} order features of images I_1, I_2 is defined as follows.

$$K_n(I_1, I_2) =< \Phi_n(I_1), \Phi_n(I_2) > . \tag{3.1}$$

To remove the bias introduced by the number of words in an image, we normal-

ize the feature vector. Thus the normalized kernel $\hat{K}_n(I_1, I_2)$ is as follows,

$$\hat{K}_n(I_1, I_2) = < \frac{\Phi_n(I_1)}{\|\Phi_n(I_1)\|}, \frac{\Phi_n(I_2)}{\|\Phi_n(I_2)\|} > \tag{3.2}$$

$$= \frac{K_n(I_1, I_2)}{\sqrt{K_n(I_1, I_2)K_n(I_2, I_2)}}. \tag{3.3}$$

Our final kernel is a weighted sum of all $\hat{K}_n(I_1, I_2)$'s.

$$K(I_1, I_2) = \sum_{n=1}^{\infty} w_n \hat{K}_n(I_1, I_2). \tag{3.4}$$

One way to set the weights w_n is using μ^{1-n} with μ a pre-defined value of $(0, 1)$. As μ gets closer to zeros, we put more weights to the higher order features.

With this kernel, the decision score of the SVM for a testing image $S(I)$ would be

$$S(I) = \sum_i y_i \alpha_i K(I, s_i) \tag{3.5}$$

$$= \sum_i y_i \alpha_i \sum_n w_n \hat{K}_n(I, s_i), \tag{3.6}$$

where s_i are the support vectors (training images) of the SVM, $y_i \in \{+1, -1\}$ are their corresponding class labels, and α_i denote the learned coefficients of the support vectors.

3.2.2 Combine HOF Kernels with MKL

We can further refine the kernel $K(I_1, I_2)$ (Equ. 3.4) by learning the weights w_n with training images. The intuition is that different objects may have different weights for different orders. Objects with rigid shapes, like *faces*, should have higher weights on higher orders, while objects with less rigid shapes, like *cat*, should have higher weights on lower orders.

The Multiple Kernel Learning (MKL) techniques [2, 115] would be suitable for this purpose. In previous work for object recognition, MKL has been applied for combining kernels computed from different types of features, such as color, texture and shape [33, 116]. We use it for combining kernel matrix of different order features.

The main idea of MKL is to formulate the SVM so that it learns the coefficient α_i (Equ. 3.6) and the weights for different kernel matrix w_n at the same time. We use the MKL technique proposed in [115] and the code provided by the authors. HOF kernels of orders 1 to 10 are combined, and a SVM with a different set of weights is learned for each class.

3.2.3 Coding Local Features

The HOF histogram in the algorithms explained above assumes that each local feature is hard quantized to a single visual word. Many papers have shown that using soft quantization improves the performance [120, 130]. In soft quantization, each local feature is assigned to a set of words with non-zero codes computed using various coding schemes, such as Sparse Coding [130] and Locality-constrained Linear Coding [120]. We show that our HOF algorithm can be easily adapted for this case.

Now since the local words are associated with weights (codes), we define the weight for a HOF as the summation of the weights of the words composing it: $c(f) = \sum_{w_i \in f} c_i$, where c_i denotes the code for word w_i. Thus, the dot product of the vector representations of two images equals to the summation of the weights of the co-occurring HOF. Suppose an offset bin of two images has N

votes generated from words $w_1, ..., w_N$; the summation of the weights of HOFs of order n in this bin can calculated as follows:

$$\sum_{f \in \{w_1,...,w_N\}^n} c(f) = \binom{N-1}{n-1} \sum_{i=1}^{N} c_i. \tag{3.7}$$

Therefore, to associate the HOF with weights, we simply modify the equation in the 2nd step of Algorithm 1 with

$$K_n(I_1, I_2) = K_n(I_1, I_2) + \binom{N-1}{n-1} \sum_{i=1}^{N} c_i. \tag{3.8}$$

3.3 Experiments

In this section, we evaluate our algorithm for the object recognition task with the public datasets: Caltech-101 [93] and Graz-01, Graz-02 [88]. We would like to verify that using Higher Order Features (HOFs) effectively encodes geometry information to the Bag-of-Words (BoW) model and improves the performance. Moreover, we compare favorably with other approaches for modeling the geometry information. For each dataset, we extract local features with the detectors and descriptors that achieve the state-of-the-art results on the corresponding dataset. These experiments also show the flexibility of our approach on different types of local features.

Practical Issue: Due to the large dimension of the feature space, images will have extremely sparse representations in the HOF feature space. This also leads to the fact that the kernel value of the same image will be much larger than that of two distinct images. Thus, our kernel matrix will be nearly diagonal, especially for higher order. This is called diagonal dominance in machine learning, and is proved to be a problem when the kernel matrix is applied to learning

algorithms such as SVM. Many methods have been proposed to overcome this problem [37]. We applied the negative diagonal shift method, which is to subtract a constant from the diagonal of the kernel matrix. Although it is possible make the kernel matrix not positive semi-definite any more, it has shown to gain good performance in practice.

3.3.1 Caltech-6

First, on a subset of the Caltech-101 dataset, we analyze some important factors of our approach. We use images of six object categories from the Caltech 101 dataset: *faces, motorbikes, airplanes, rear cars, watches, ketches*. We apply Harris-Laplacian interest point detector [83] and SIFT descriptors [77]. We report classification results for two classes (faces and motorbikes), four classes (faces, motorbikes, airplanes, and rear cars), five classes (4 classes + watches), and six classes.

Kernel Matrix

Figure 3.1 shows the kernel matrix of the training data with five classes for different order features. As the order increases, the matrix gets darker (values are smaller) at the off-diagonal places, and thus is more discriminative among different categories. However, the diagonal places (same category image pairs) also get darker for some categories, such as airplane and rear car, with higher orders (7 or 10). This is due to the fact that few co-occurring higher order features are detected for some image pairs in the same category when the category includes more variance in object shapes. Faces and Motorbikes are two objects with the most consistent structures.

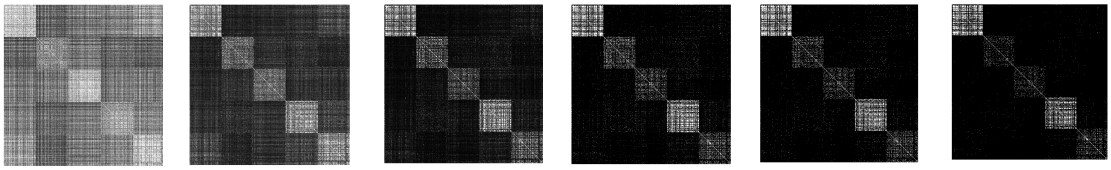


Figure 3.1: Kernel Matrix of the training images for different order features. From the left to right are 1st, 2nd, 3rd, 5th, 7th, 10th order kernel matrices. Training images are arranged in the categories of *face*, *airplane*, *rear car*, *motorbike*, and *watches*.

Individual vs. Cumulative

We show the classification results with K Nearest Neighbor classifier (k=3) in Figure 3.2. We use the normalized kernel as the similarity function for KNN. We present both the results of individual kernels K_n with only the n^{th} order features, and the results of the sum of kernels from 1st to n^{th} order individual kernels. We show the performance until 10^{th} order since most images do not have co-occurring features whose order are larger than 10. We found that for both individual and cumulative kernels, we gain significant improvement from 1st order (bag of words) to 2nd order features, which proves that modeling geometry information helps a lot. For the individual case, we get best performance with $8th$ order kernel for 2 class classification task, 2nd for 4 class, and 3rd for both 5 and 6 class. The accuracy dropping is mainly because as the order increases, fewer images will have co-occurring features as we have seen in the kernel matrix (Figure 3.1). For the cumulative case, the accuracy generally keeps increasing as we increase the order (may stop growing at some order). We found that even if individually the accuracy drops for the higher order kernels, they may still contribute to the performance when we add them together. This is not surprising given the fact that higher order features are more discriminative. We reach best performance when the maximum order is 10 for 2 class, 5 for 4, 5, and 6 class

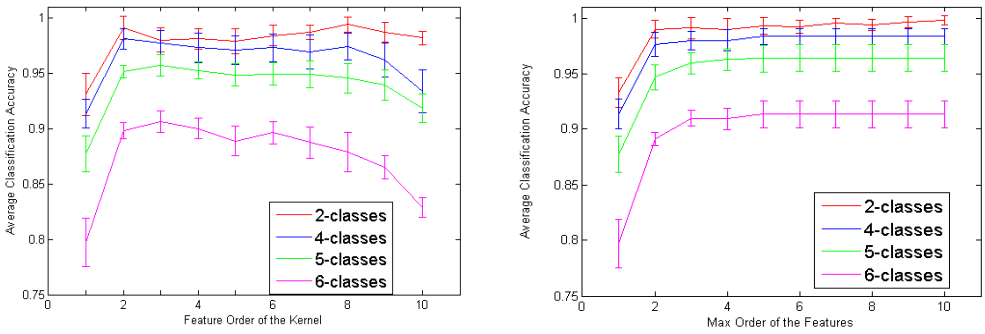


Figure 3.2: Object categorization performance with KNN (k=3). We show the average classification accuracy for classification task with 2, 4, 5, and 6 classes. The x axis is the feature order n. The left figure shows the accuracy only using the n^{th} order kernel. The right figure shows the accuracy using the summation of the kernels from 1 to n.

classification tasks.

Using as Kernel for SVM

We apply the kernel to the SVM. Figure 3.3 shows the results for the sum of the individual kernels. The accuracy with SVM is better than that with KNN for all the orders. We can still see the performance increase as we increase the order.

Vocabulary Size

The experiments till now are using a vocabulary size 500. We now decrease the vocabulary size to 50. Figure 3.4 shows the classification performance when using KNN and SVM. In this case, the individual visual words would be more ambiguous, which results in low accuracies when using low order features. Especially for KNN, the accuracy is under 70% when using bag of words model. However, for both KNN and SVM, as we increase the order of the features, we got the performance quite close to that when we use 500 visual words. This

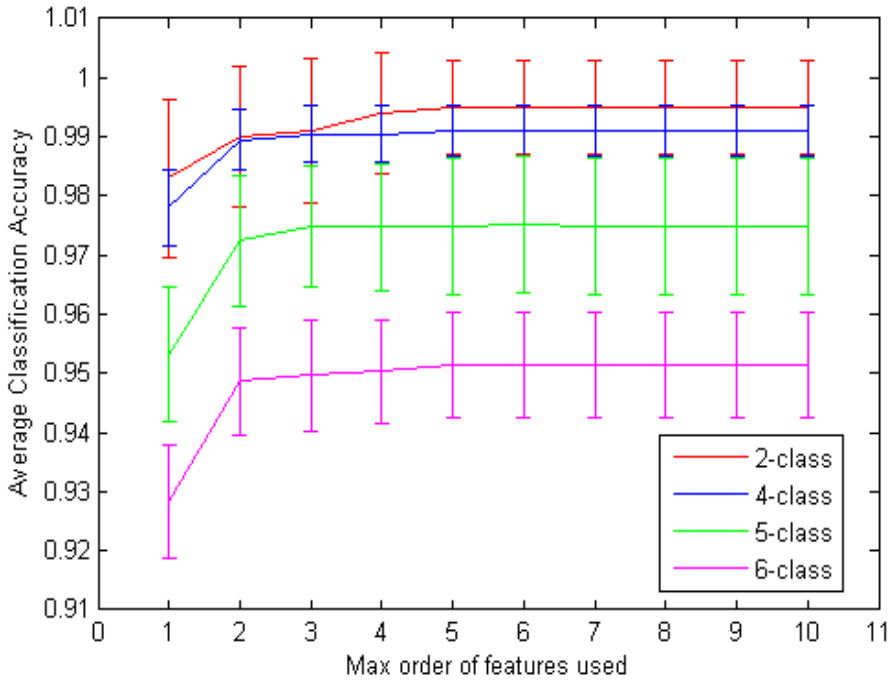


Figure 3.3: Object categorization performance with SVM. We use sum of the individual kernels.

shows that even if the local features are not discriminative itself, by increasing the orders, we can create discriminative high order features.

Running Time

We run our experiments on a *single CPU* of a 2.26G Quad-Core Intel Xeon server with 12G memory. Computing the kernel values for a pair of images takes 0.2ms with a Matlab+C implementation.

3.3.2 Caltech-101

The whole Caltech-101 dataset [93] contains 9144 images of 101 object categories. Each category includes 31 to 800 images. Following previous work

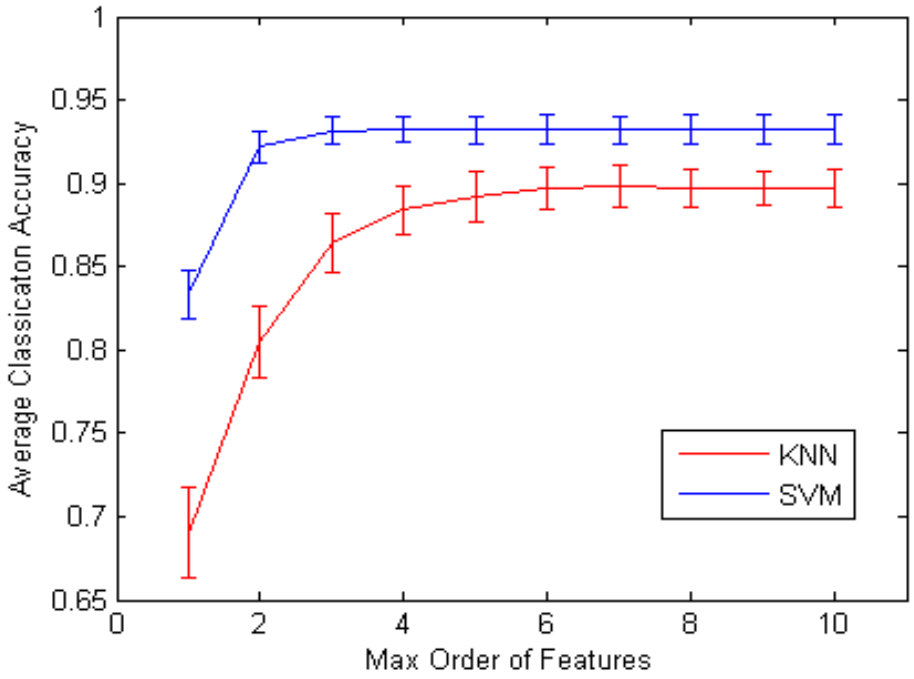


Figure 3.4: Categorization accuracies with vocabulary size 50. We show the results for 6 class classification.

[93,115,120], we train the SVM classifiers by randomly selecting 5,10,15,20,25,30 images per class and test with no more than 50 images. Experiments are repeated 5 times and the average accuracies are reported.

In this experiment, we use the same feature detector and descriptor as [120]: local patches of size 16×16 are extracted densely at every 6 pixels on the image and represented with the Histogram of Oriented Gradient (HOG) descriptor. The same coding scheme (Locality-constrained Linear Coding) [120] is applied to create the visual words and their codes. We use a vocabulary size of 1024 learned with Kmeans, and encode each local feature to its 2 nearest neighbor words. Previous work [120, 130] have shown that a max-pooling step on the codes of the descriptors improves the performance. For our implementation, we also max-pool the sub-regions of a 2×2 grid, and apply our HOF algorithm

on the pooled local features. We found that this step is important when *dense* features are used, where many nearby features are assigned to the same word, and thus create unreliable HOFs.

Word Encoding

Table 3.1 compares our approach with Bag-of-Words (BoW), Spatial Pyramid Matching (SPM) L2 (level 1-2), and SPM-L3 (level 1-3). The same pipeline as [120] is used for SPM. For SPM, level 2 and 3 uses 2×2 and 4×4 grid respectively. We show the average accuracies and standard deviations over all 102 categories when 15 training images are used. Performance using different word encoding methods are reported. For words obtained with hard vector quantization (VQ), we use χ^2 kernel for both BoW and SPM, which performs better than a linear kernel. While for words obtained with Locality-constrained Linear Coding (LLC) [120], we use linear kernels for both BoW and SPM, since χ^2 kernel performs worse than the linear one in this case. Our methods use the proposed HOF kernel in both cases.

The table shows that HOF outperforms other methods in both average accuracy and standard deviation no matter which coding method is used. Moreover, HOF can be successfully combined with LLC coding to further improve the classification accuracies.

Combine HOF Kernels

Table 3.1 also shows that using the learned weights for combining different order kernels (HOF-learned) improves the performance of the method that manually sets fixed weights for all categories.

Table 3.1: Average accuracies on Caltech 101 dataset. Classifiers are trained with 15 images per class. For different methods, we show the performance using different word encoding method: hard vector quantization (VQ) and Locality-constrained Linear Coding [120]. We compare our method (HOF) with BoW, Spatial Pyramid Matching (SPM) of level 1 and 2 (2×2), and SPM of level 1 to 3 (4×4). HOF-learned use the learned weights for combining different order kernels.

	BoW	SPM - L2	SPM -L3	HOF	**HOF-learned**
VQ	44.26±2.24	51.2±1.12	57.09±0.89	61.87±0.33	**62.94±0.35**
LLC	37.18±1.95	52.2±0.68	64.55±0.56	67.17±0.36	**68.28±0.35**

Figure 3.5 shows the learned weights of HOF kernel of different orders averaged over all categories. For the multiple kernel learning, we use $l1$ norm for the weights of different orders, which favors sparsity of the weights. Interestingly, all categories give zero weight for the first order kernel (BoW). The reason is because all object categories in Caltech-101 are in some particular shape (although some are more rigid), and thus combining the orderless word based kernel with other HOF kernels do not help. Second order feature gets the highest weight in general. Many categories give zero weights to 6-8th order features, which is probably because the information of these order features can be captured by other orders.

In figure 3.6, we present the learned weights of HOF kernels (order 1 to 10) for categories that have lowest weights (left) and highest weights (right) for the 10^{th} order. As we expected, the objects, which are more rigid in shape and more consistent in appearance, get higher weights for the higher orders. For lower orders, the opposite trend is shown.

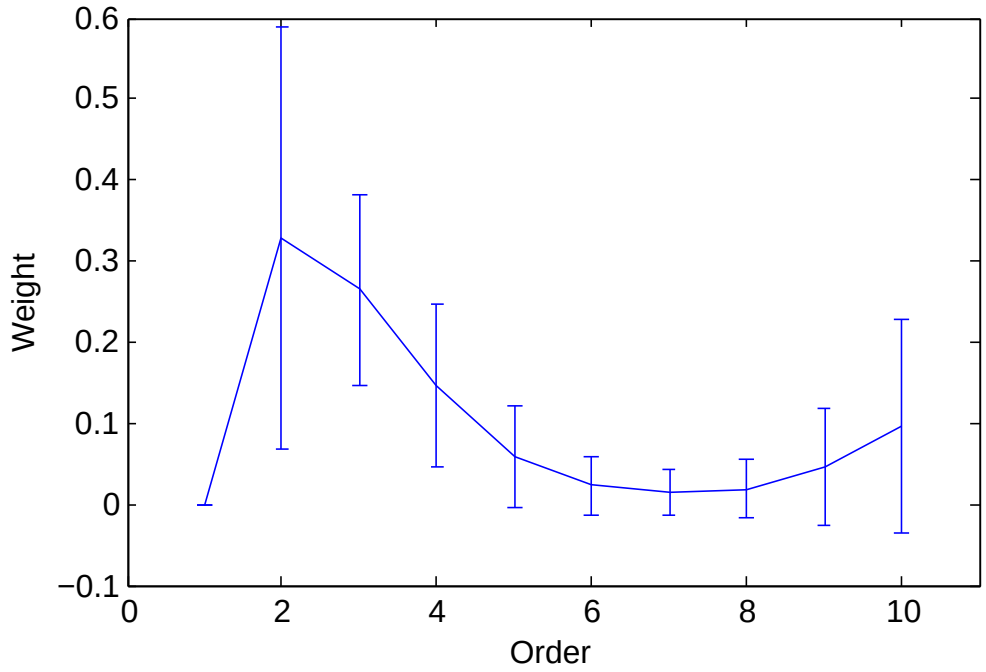


Figure 3.5: Learned weights for HOF kernel of different orders. We show the average and standard deviation over the learned weights of all categories.

Offset Space Quantization

In our algorithm for computing the co-occurring HOFs of two images (Algorithm 1), we consider any N votes that fall in the same bin of the quantized offset space as a co-occurring N^{th} order feature. Figure 3.7 shows the average accuracies when different step sizes are used for the quantization. Intuitively, smaller step size will work better for objects of more rigid shape. Our final kernel combines the kernels computed with step size 5,10,15,20,30, and achieved the better result.

Comparison

Figure 3.8 compares our approach with BoW and SPM under various number of training images per category. Exactly the same local features and the

same coding method (LLC) are used for these methods. Therefore, the figure proves that our HOF based approach outperforms the performance of BoW by incorporating geometry information among the local words. Moreover, our approach also outperforms SPM which is a popular method for encoding spatial information to BoW.

Finally, we compare the performance of our approach with other recent work that achieve the state-of-the-art results on the Caltech-101 dataset in table 3.2. Methods cited in the top rows of the table use similar local features (dense HOG/SIFT) and similar coding methods as us, but different ways for encoding the geometry information. ScSPM [130] and LLC+SPM [120] uses Spatial Pyramid Matching (SPM); RLDA [47] and Receptive Field [45] improves SPM by improved algorithms for creating the bins of the image space. The bottom rows of the table present the performance of other methods that either use different type of features (SPM [64], NBNN [6], Deconv. Net [135], LP-B (MKL) [33]), different pooling method (Deconv. Net [135]), or different classifiers (NBNN [6], Deconv. Net [135]). In particular, LP-B (MKL) [33] uses Multiple Kernel Learning to combine many different types of features, such as texture, color, shape, and self-similarity. GLP [28] proposes a different pooling (feature selection) method other than max-pooling and achieves the best result on this dataset. The improvement of these methods is orthogonal to ours, since our approach can be applied to any local features and can always be plugged in after the pooling step to capture the geometry relationship of the words remained.

Table 3.2: Average accuracy (%) comparison on Caltech-101. Top: methods that use the similar local features as ours. Bottom: other recent work that achieve good results on Caltech-101.

Training images	5 train	15 train	30 train
ScSPM [130]	-	67	73.2
LLC+SPM [120]	51.15	65.43	73.44
RLDA [47]	-	-	73.7
Receptive Field [45]	-	-	75.3
Ours (HOF-learned)	55.14	68.28	77.23
SPM [64]	-	56.4	64.6
NBNN [6]	-	65	70.4
Deconv. Net [135]	-	-	71
GLP [28]	59.35	70.34	82.6
LP-B (MKL) [33]	54.2	70.4	77.7

3.3.3 Graz-01, Graz-02

The Graz-01, Graz-02 datasets [88] include objects of various scales. Objects in these datasets have large location and scale differences; therefore, methods that are not translation and scale invariant, such as Spatial Pyramid Matching, are not suitable. Figure 2.5 shows example images from the datasets. Following previous work [72] on these dataset, we use harris-hessian interest region detectors [82] and SIFT feature descriptor [77]. We build the vocabulary with K-means using $K = 500$. Translation and scale invariance are considered during the experiments. We adopt the same training and testing split as in [88].

Table 3.3 compare the Equal Error Rates (EER) of each category with pre-

Table 3.3: Equal Error Rate (%) on the Graz dataset

Dataset	Class	BoW [88]	Pair [67]	PDK [72]	HOF
Graz01	Bicycle	86.5	84.0	95.0	**96.0**
	Person	80.8	82.0	88.0	**90.0**
	Bike	77.8	**92.0**	86.7	88.0
Graz02	Person	81.2	86.0	86.7	**92.0**
	Car	70.5	n/a	74.7	**80.3**

vious works. We outperform other methods which use bag of words [88], and methods that encode geometry information with pair of words [67] [72] on most categories.

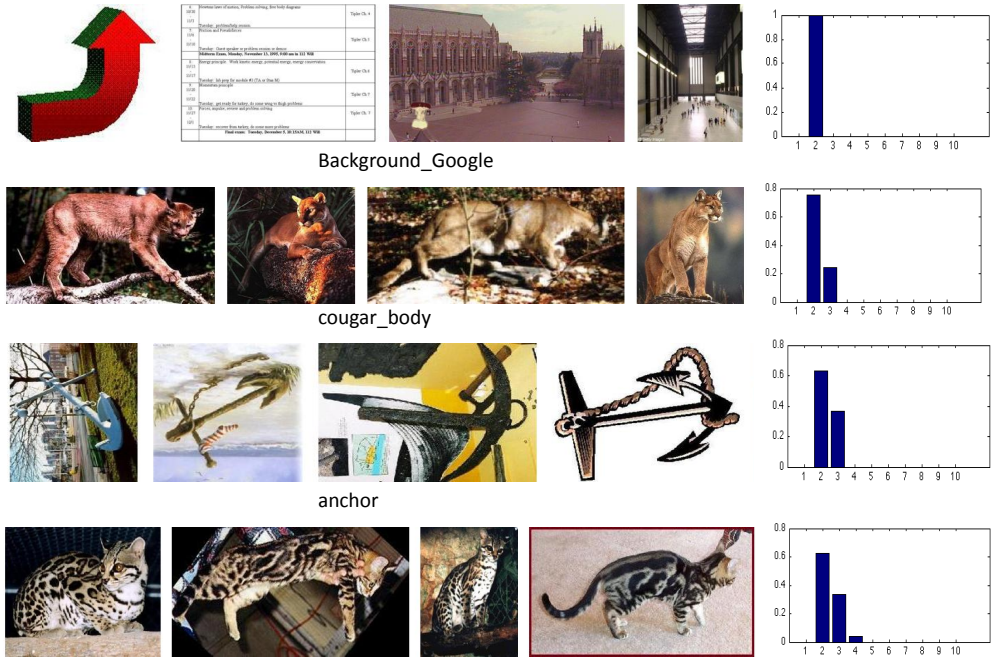


Figure 3.6: Learned weights of HOF kernels of different orders. We show example categories that have lowest weights (a) and highest weights (b) for the higher (10th) orders. For each example category, example images and the learned weights for orders 1 to 10 are shown.

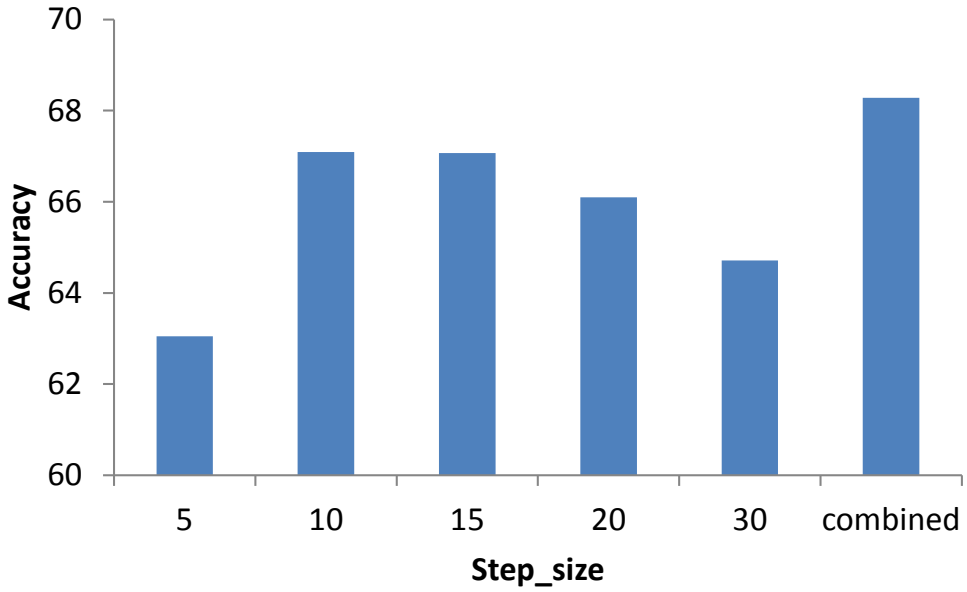


Figure 3.7: Average Accuracies when different step sizes are used for quantizing the offset space. 15 images per category are used for training.

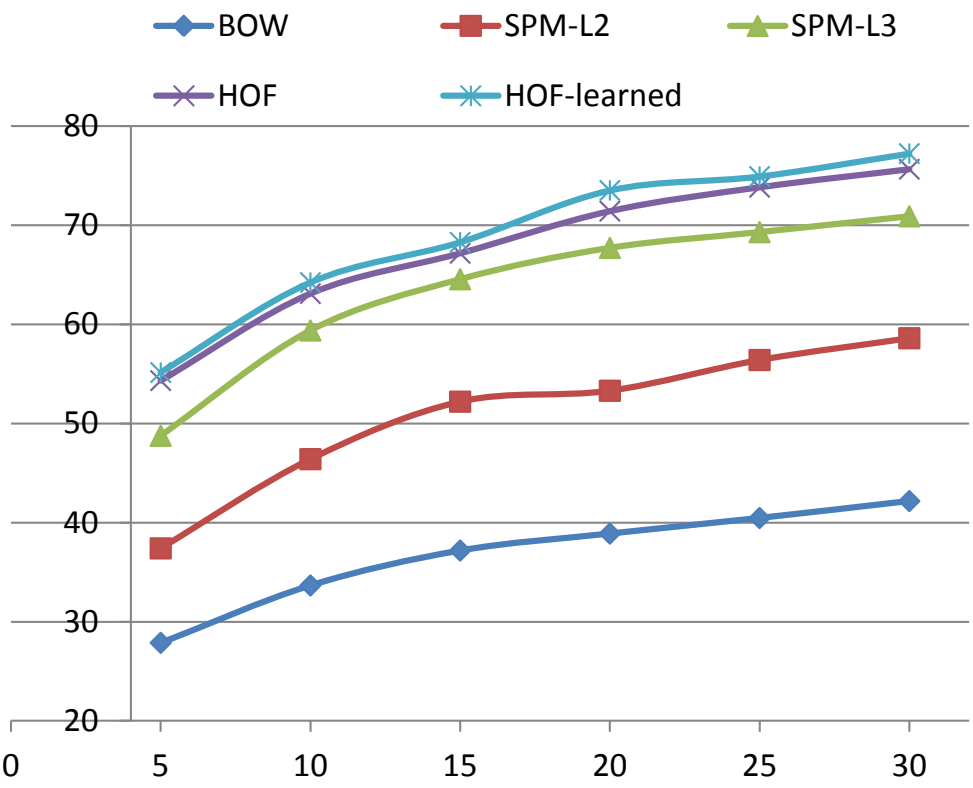


Figure 3.8: Average Accuracies of different methods when different number of images per category are used for training.

CHAPTER 4

HOF FOR OBJECT DETECTION

The task of object detection not only recognizes whether an object is in an image, but also detects the location of the object. In this chapter, we introduce how to efficiently adopt the proposed HOF algorithm for this task.

A naive way to perform object detection with the HOFs is the sliding window approach, which performs classification for every possible sub-windows in an image. Classification can be done in the same way as "object recognition" (Chapter 3) with the HOF kernel for the SVM. However, this method would require computing kernel values for a large number of sub-windows per image. We propose an efficient algorithm which obtains the decision scores for all sub-windows with only a single kernel calculation for the entire image.

Moreover, we present the method that integrates the HOF algorithm into the structured learning framework. Unlike the sliding window approach, which learns a binary classifier for the sub-windows in an image, the structured learning can formulate the output of the classifier as the location of the object of interest. Thus the training of the classifier is directly optimizing the localization performance.

4.1 Related Work

The sliding window approach [9, 18, 25, 30] been widely used for object detection. During training, a binary classifier which determines the presence of the object of interest in a sub-window is trained with a sample of positive and neg-

ative examples. During testing, the classifier is evaluated at every possible location and scale in a test image to localize the object in the image. Despite the effectiveness of the approach, one main disadvantage is that the training is not directly optimizing the detection performance.

Structured learning has been used to address the problem of the sliding window approach [4, 5]. Rather than predicting a binary label, the classifier with structured learning is learned to predict a more structured output, which is the bounding box of the object for the object detection task. A bounding box is parameterized as the top, left, bottom and right coordinates of the box. Thus the output space of the object detection task can be represented with the four numbers. Thus, the classifier is trained to directly optimize the detection performance.

One important issue with structured learning is the efficiency for the inference, since the size of the output space is quite large. To learn a structured classifier, we usually need to iteratively perform inference on the training data to find the negative examples. To efficiently find the optimal bounding box in an image, previous works use the branch and bound algorithm with the Bag of Words (BoW) representation [4, 59]. However, BoW representation does not model the shape of the object, and thus is not discriminative enough. We incorporate the HOFs to the structured learning framework to improve the performance of BoW, and provide an efficient sub-window search algorithm when the HOFs are used.

4.2 Object Detection with HOF

4.2.1 Structured Learning Review

The outputs of a classifier learned with the structure learning [113] are not simple binary labels, but instead have a more complex structure. This allows us to better model the relationship between different outputs within the output space. In order to formulate object detection as a problem of predicting structured data, we follow the framework proposed in [4]. We briefly introduce the framework here.

In the context of object detection, the problem is defined as below: given a set of input images $\{x_1, \cdots, x_n\} \subset \mathcal{X}$ and their associated object annotations $\{y_1, \cdots, y_n\} \subset \mathcal{Y}$, we wish to learn a mapping $f : \mathcal{X} \rightarrow \mathcal{Y}$ with which we can automatically annotate unseen images. The output space consists of two parts: a label indicating whether an object is present, and a vector indicating the top, left, bottom, and right of the bounding box within the image: $\mathcal{Y} \equiv \{(\omega, t, l, b, r) | \omega \in \{+1, -1\}, (t, l, b, r) \in \mathbb{R}^4\}$. $\omega = -1$ indicates no object is present. So the coordinate vector (t, l, b, r) is ignored in this case. The mapping from $\mathcal{X}$ to $\mathcal{Y}$ is learned in the structure learning framework [113] as

$$f(x; \omega) = \arg\max_{y \in \mathcal{Y}} F(x, y; \omega) \tag{4.1}$$

where ω denotes a parameter vector and $F(x, y; \omega)$ is a discriminant function. We further assume F to be linear in some feature representation of inputs and outputs $\psi(x, y)$,

$$F(x, y; \omega) = < w, \psi(x, y) > \tag{4.2}$$

The feature representation $\psi(x, y)$ will be defined in Section 2.3.2. To train the

discriminant function, $F(x, y; \omega)$, we use the structured SVM method proposed in [113]

$$\min_{\omega,\xi} \frac{1}{2}\|\omega\|^2 + C\sum_{i=1}^{n}\xi_i \tag{4.3}$$

$$\text{s.t. } \xi_i \geq 0, \forall i \tag{4.4}$$

$$\langle\omega, \psi(x_i, y_i)\rangle - \langle\omega, \psi(x_i, y)\rangle \geq \Delta(y_i, y) - \xi_i, \forall i, \forall y \in \mathcal{Y} \setminus y_i \tag{4.5}$$

where

$$\omega = \sum_{i=1}^{n}\sum_{y\in\mathcal{Y}\setminus y_i} \alpha_{iy}(\psi(x_i, y_i) - \psi(x_i, y)) \tag{4.6}$$

and $\Delta(y_i, y)$ is a loss function chosen to reflect the quantity that measures how well the localization performs. In this work, the loss function $\Delta(y_i, y)$ is defined as Equation 4.7.

$$\Delta(y_i, y) = \begin{cases} 1 - \frac{Area(y_i \cap y)}{Area(y_i \cup y)} & \text{if } y_{i\omega} = y_\omega = 1 \\ 1 - (\frac{1}{2}(y_{i\omega}y_\omega + 1)) & \text{otherwise} \end{cases} \tag{4.7}$$

The loss function has the following properties: 1) it is equal to 1 when the labels of the bounding boxes disagree; 2) it is equal to 0 when the labels of the bounding boxes are both negative; 3) it is measured by the area overlap between the boxes when the labels of the bounding boxes are both positive. $\Delta(y_i, y)$ is 1 if the boxes are identical and 0 if they are disjoint.

The key problem for solving the generalized SVM learning is the large number of margin constraints defined in Equation 4.5. Following the methods proposed in [113], we here use a cutting plane method to find a subset of active constraints that ensures a sufficiently accurate solution. It is equivalent bewteen Equation 4.5 and

$$\xi_i \geq \max_{y\in\mathcal{Y}\setminus y_i} \Delta(y_i, y) - (\langle\omega, \psi(x_i, y_i)\rangle - \langle\omega, \psi(x_i, y)\rangle), \forall i \tag{4.8}$$

In training, we iteratively repeat the following: 1) estimate ω using the fixed subsets of constraints; 2) add new constraints by finding y that maximize the right part of Equation 4.8. In testing, we find y that maximizes Equation 4.1.

4.2.2 High Order Feature in a Sub-window

Let x denote an image, y denote a sub-window, and $\phi^n(x, y)$ denote the n^{th} order feature vector for this pair of input and output. The values of the feature vector of a sub-window y are defined as the number of occurrences of the n^{th} order features in y. When counting the occurrence of a n^{th} order feature, we count one when all n patches of the feature are inside the sub-window y, otherwise, we count k/n, where k is the number of patches inside y. That is, we take into account the context of a sub-window when making its prediction. Context has been proved to improve the detection performance by many previous works [5, 39, 89].

4.2.3 Inference Algorithm

We start by explaining the localization/inference algorithm. A similar algorithm can be used to efficiently calculate the kernel values of the training images to train a SVM. We will talk about the training in later section. Suppose we have already learned a SVM, that is, we have the coefficients α for the support vectors.

The key idea for efficient inference is to avoid computing the feature vector and avoid calculating kernels for every possible windows in a test image. We

first assign scores to the local patches in the test image. We define the scores in the way that the SVM decision value for a subwindow can be calculated as the summation of the scores of the local patches inside the window. Thus after obtaining the scores of the local patches, we can easily get the optimal subwindow with the branch and bound algorithm.

Local patch score

The score for a local patch p_i is defined as the summation of the learned weights of all n^{th} order features including that patch. Let $I^n_{p_i}$ denote all n patches including p_i, and $\phi^n(I^n_{p_i})$ denote the n^{th} order feature vector of $\phi^n(I^n_{p_i})$, which is the number of occurrences of the features. The score of a patch is calculated as:

$$Score(p_i) = \mathbf{v}^T \phi^n(I^n_{p_i}) \tag{4.9}$$

where $\mathbf{v}$ is the weights of the n^{th} order features of the SVM. It is not efficient to list the weights for all high order features, we use the kernel methods. The weight v_k for a n^{th} order feature f^n_k can be computed as:

$$v_k = \sum_t \alpha_t \phi^n_k(x_t, y_t) \tag{4.10}$$

where α is the coefficient for the support vectors, and $\phi^n_k(x_t, y_t)$ is the value for the k^{th} coordinate of the feature vector for the support vector (x_t, y_t). A support vector is a certain training example, which is represented as the image x_t and the subwindow y_t. The feature value is the number of occurrences of a feature in the subwindow y_t as defined in Section 4.2.2.

Thus, combining equation 4.9 and 4.10, the scores of the patches in an image can be calculated by identifying the co-occurring HOFs between the image and each support vector. We use the proposed algorithm (Chapter 2) to efficiently detect the co-occurring HOFs. Figure 4.1 illustrates the algorithm. With each

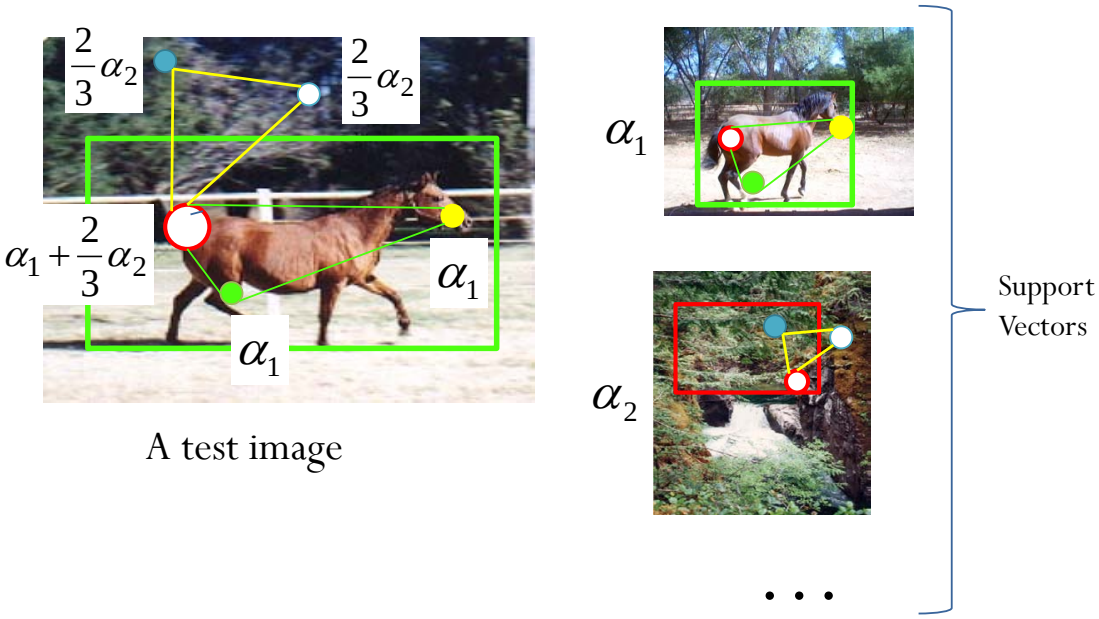


Figure 4.1: The illustration of algorithms for calculating the scores of local patches. The circles are the local patches. Different color represents different word assignments.

support vector (x_t, y_t), we identify the co-occurring n^{th} order features of the test image. When a co-occurring feature is found, we increment the scores of the patches of the test image that compose this feature by $\alpha_t \frac{l}{n}$, where l is the number of patches out of the n patches inside the subwindow y_t of the support vector.

Subwindow scores

It can be easily seen from the following that the decision score of a subwindow y of a test image x can be calculated as the summation of the scores of the

patches inside it.

$$
\begin{aligned}
f(x,y) &= \sum_t \alpha_t K_n((x,y),(x_t,y_t)) + b && (4.11)\\
&= \mathbf{v}^T \phi^n(x,y) + b && (4.12)\\
&= \frac{1}{n}\sum_{p_i \in B} \mathbf{v}^T \phi^n(I^n_{p_i}) + b && (4.13)\\
&= \frac{1}{n}\sum_{p_i \in B} Score(p_i) + b && (4.14)
\end{aligned}
$$

where b is the bias term of the SVM.

Therefore, after assigning the scores to every patch in an image, the process of finding the optimal subwindow would be the same as the process of the bag of words model with a linear SVM, while the weights of the words in the bag of words model are replaced by the scores of the patches in our algorithm. Therefore, we can use the branch and bound algorithm with the upper bound proposed for the bag of words in the paper [59].

4.2.4 Learning

The inference algorithm can also be used for efficiently calculating the kernel value between two training examples (x, y) and (x', y'). We calculate the scores for the local patches in image x with the algorithm in Section 4.2.3 by assuming that there is only one support vector (x', y'), and the $\alpha = 1$. Then we add up the scores of patches inside y, and this will give us the kernel value between the two examples. This is much faster than directly calculating the inner product of the feature vectors of the two examples, especially when n is large, since the feature vector can be impractically long with n larger than 2.

4.3 Experiment

We evaluate the proposed algorithm on the INRIA horse dataset [39]. The dataset consists of 170 images with one or more side-views of horses and 170 images without horses. We randomly select 50 images with horses and 50 images without horses for training. From the remaining images, we randomly select the same number of images for validation to decide the SVM's regularization parameter C. The rest of images are used for testing. We use visual words (clusters of SIFT features [78]) and the corresponding high-order combination of the words to represent an image. The visual code book contains 2000 words.

The structure learning algorithm is implemented with the SVM^{light} package [46]. We solve the problem by adapting the cost input to the loss function defined in Equation 4.7 and adapting the kernel function to our high-order feature kernel.

To evaluate the detection performance, we use one of the standard object detection metric: Recall-FPPI curve. Recall is the detection rate of all positive boxes. FPPI (False Positive Per Image) is the average number of false positive boxes on an image. Outputs of the algorithms are the predicted bounding boxes in images with confidence scores associated. For a predicted bounding box B_P, we consider it as a correct detection, if the area of overlap between the predicted bounding box B_P and the ground truth bounding box B_T is over 50%. The overlap is calculated as follows, consistent to which we use in the loss function.

$$overlap = \frac{area(B_P \cap B_T)}{area(B_P \cup B_T)} \tag{4.15}$$

We conduct two experimental comparisons. First, we compare the detection

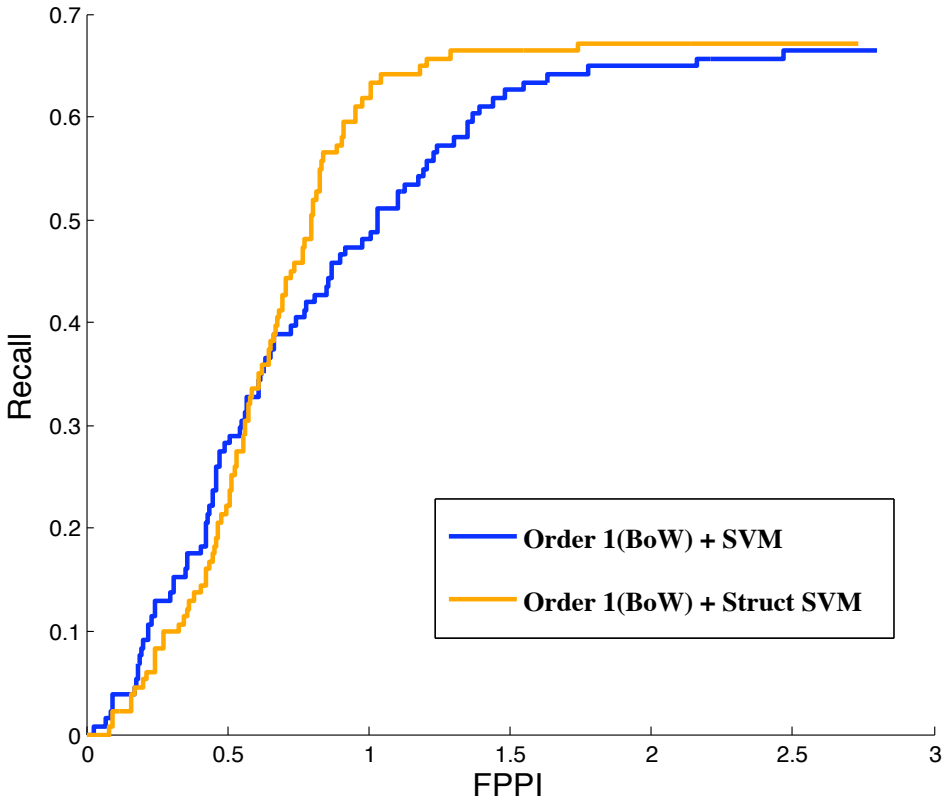


Figure 4.2: The performances achieved by using the linear SVM method and by the structured SVM method, both with bag-of-words features.

performance of using structure learning method with that of using a binary linear SVM classifier. In this case, we use the bag-of-words features [17], that is, $\psi(x, y) = \phi^1(x, y)$. This is the same set-up in [4] when comparing the structure learning method with the binary classifier method. Results are given in Figure 4.2. We can see that based on the same features, the structure learning method achieve better performance than the binary classifier in most cases, whie at low FPPI region the linear SVM method gives slightly better result.

Secondly, we compare the detection performances of features at different orders, that is $\psi(x, y) = \phi^i(x, y)$, with $i \in \{1, 2, 3, 4\}$. This group of experi-

ments are all conducted with the same structure learning method. We also try to utilize features at all these four orders by appending them together: $\psi(x,y) = [\phi^1(x,y) \;\; \phi^2(x,y) \;\; \phi^3(x,y) \;\; \phi^4(x,y)]$. We test the cases of using a single order feature and using the multi-order features. Results are given in Figure 4.3. We can see that the performance increases when we increase the feature order from 1 to 4. We did not try even higher order features for very few patches with order higher than 4 can be found on the images. The combined order features achieve similar result as that of the 4^{th} order, and performs especially better at the low FPPI region.

We show the example detections by our approach using the features from order 1 to 4 in Figure 4.4.

Computation Time

It takes averagely 80 seconds for training on 100 images and 0.3 second per image in testing. The algorithm is implemented in Matlab with some C helps. The experiments are conducted on a Mac OS X system with a 2.4GHz CPU and 2GB memory.

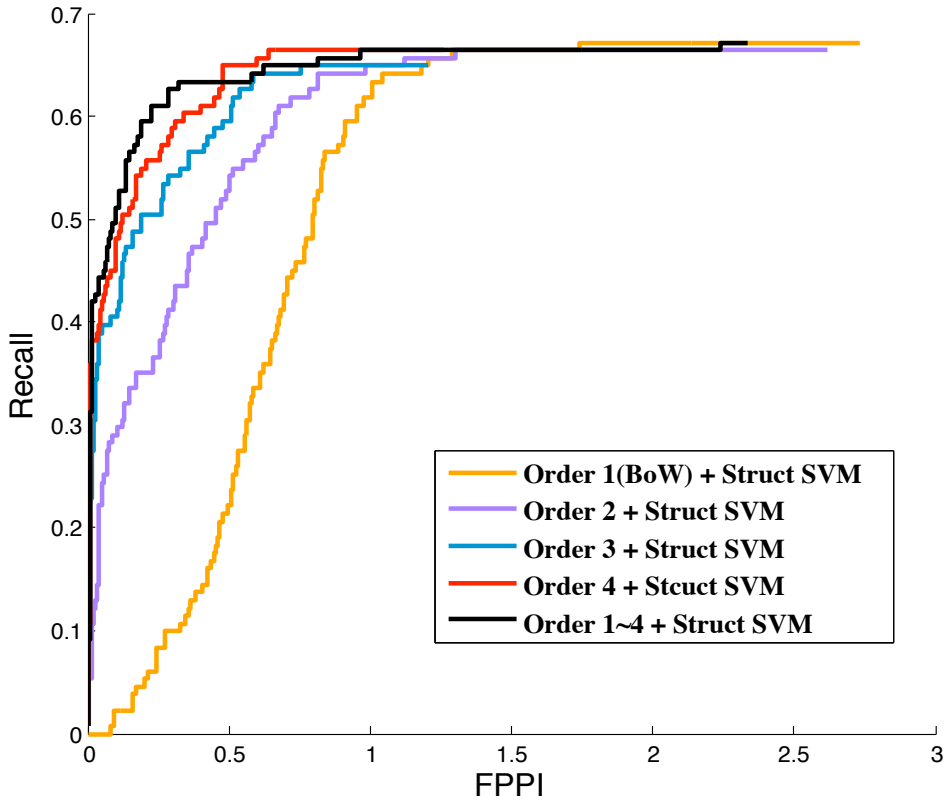


Figure 4.3: The performances achieved by features at different orders. The black line is achieved by combining all four order features. The rest are achieved by using a single order feature. All the results are achieved from the structure SVM method.

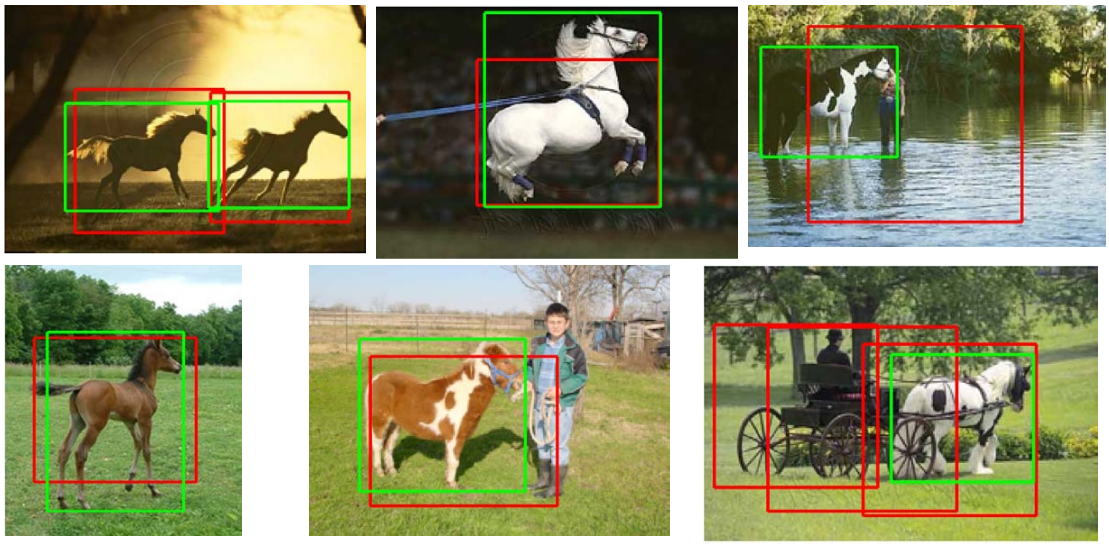


Figure 4.4: Example detections. The green rectangles are the ground truth bounding boxes, and the red ones are the predicted bounding boxes by our algorithm.

CHAPTER 5

HOF FOR IMAGE RETRIEVAL

Similar image retrieval has attracted increasing interests in recent years. Given a query image or region, the goal is to retrieve the images of the same object or scene from a large database and return a ranked list. Three important factors must be considered in a large-scale retrieval system: retrieval accuracy, memory usage, and efficiency.

5.1 Related Work

Most state of the art retrieval technologies are based on the bag-of-word (BoW) model initially introduced by [106], in which images are represented as histograms of visual words. Image querying is typically accomplished in two steps: *searching* and *post-processing*. During the *searching* step, similar images are retrieved from the large database and an initial ranking is generated. This step must be facilitated with an efficient algorithm in order to deal with large scale databases. The most popular approach is to index images with inverted files [106] to facilitate fast access to the images with common words. The *post-processing* step provides a more precise ranking of the retrieved images, usually through spatial verification [94]. Numerous works have been proposed and have successfully improved the retrieval performance and efficiency. The approximate nearest neighbor [94] and tree vocabulary [85] increase the efficiency of building a large vocabulary, while soft matching [95] and hamming embedding [42] address the hard quantization problem of visual words. Spatial verification methods [94] and query expansion [15] have been proposed for

re-ranking at the post-processing step, and many methods [43, 44, 92, 138] have been introduced to decrease the memory usage for the inverted files.

Despite its simplicity and efficiency, the BoW model discards spatial information, which is crucial for visual representation because of the ambiguity of visual words. Spatial information is usually re-introduced in the post-processing step through a geometry verification, such as RANSAC [94] or neighboring feature consistency [106]. Since geometry verification methods are usually computationally expensive, they are applied only to the top images in the initial ranking. Therefore, efficient algorithms that encode more spatial information into the searching step are beneficial. Lin and Brandt [71], and Lampert [60] rank the images based on the matching scores of the query image with the localized sub-windows in images. These methods encode more spatial information than the the BoW model on the entire image and provide localizations of the query. However, when the query region is large, they are used primarily as a post-processing step because of the memory usage and speed [71]. Spatial Pyramid Matching (SPM) [65] and methods with GIST features [125] encode rigid spatial information by quantizing the image space and lack the invariance to transformations. Spatial-bag-of-features [8] handle variances of SPM by changing the order of the histograms; the spatial histogram of each visual word is rearranged by starting from the position with the maximum frequency, resulting in improvement over both BoW and SPM. However, this rearrangement may not correspond to the true transformation.

Another approach is to search using phrases or collocations generated from visual words, which correspond to the high-order features. Previous works usually use the phrases to model the co-occurrences of the words, either in the entire

images or in local neighborhoods. Co-occurrences in the entire images [111] do not capture spatial information between the words, while co-occurrences in local neighborhoods [129, 133, 140] capture neighboring information, but ignore long-range interactions. Moreover, they ignore the spatial layouts of the words in the neighborhoods [133, 140] or only perform *weak* spatial verification [129]. Another problem with existing methods [111, 133, 140] is that, because the total number of HOFs can increase exponentially to the number of words in a HOF, they must select a subset from the entire HOF set. Sophisticated mining or learning algorithms have been proposed for this selection, but it may still be risky to discard a large portion of HOFs, some of which may be representative ones for the images.

Moreover, our approach can integrate the HOF into the popular min-hash method to further improve the efficiency of searching with HOF, because the min-hash method reduces memory usage and increases the search efficiency. The traditional min-hash method [14, 16] is based on the BoW model. Our approach increases its retrieval accuracy by adding spatial information without increasing the computational cost. In this line of work, we are related to [13] and [12]. While they consider local co-occurrences [13] or global co-occurrences [12], we encode more spatial information.

5.2 HOF with Inverted Index

We define the similarity between two images as the cosine similarity of their HOF histograms. However, since we need to search a large amount of images, we cannot afford computing the similarities of the query image with all images

in the database. Therefore, we integrate the proposed algorithm with the inverted files method, which has been widely used with BoW for the retrieval task [106].

5.2.1 Inverted Index with BoW

We first review the traditional searching scheme with the inverted file structure [106]. An image I_i is represented as a vector $V(I_i)$, with one component for each visual word in the dictionary. The j^{th} component in the vector $v_j(I_i)$ is the weight of the word j: the tf-idf weighting scheme is usually used. The similarity of two images I_i and $I_{i'}$ is defined as the *cosine similarity* of the two vectors $V(I_i) \cdot V(I_{i'})/(||V(I_i)||||V(I_{i'}||)$. Ranking with this similarity also gives the same result as ranking with the Euclidean distance of the L2-normalized vectors. With a large vocabulary, this vector representation is very sparse. The inverted file structure [106] utilizes this sparseness to index images and enables fast searching. For each visual word in the vocabulary, this structure stores the list of images in which the word occurs and its term frequency (tf).

Searching Scheme: Given a query image q, the search can be interpreted as a voting scheme [42]: 1) The scores of all images in the database are initialized to 0. 2) For each word j in the query, we retrieve the list of images that contain this word through the inverted files. For each image i in the list, we increment its score by the weight of this word $score(i) += tf_{ij} \times idf_j$. After processing all words in the query, the final score of image i gives the dot product of the vectors of image i and the query. 3) We normalize the scores to obtain the cosine similarities for ranking.

5.2.2 Searching with HOF

Now we introduce the method that integrate the proposed HOF algorithm (Algorithm 1) in Section 2.3 with inverted files. Suppose for each visual word in the inverted files, other than the images that contain this word, we have also stored the location in which the word occurs. Multiple entries are created if a word occurs multiple times in one image. We will discuss about the storage strategy in the later section.

The key idea is to efficiently calculate the offset space for every image in the database during the process of searching. We modify the searching scheme introduced in the previous section. Rather than keeping one bin for each image for accumulating the scores, we keep M bins for each image, where M is the number of possible offsets. The voting procedure for obtaining the similarity scores of n^{th} order feature is as follows.

1. Initialize M bins for each image in the database to 0. Each bin represents an offset value.
2. For each word j in the query image, retrieve the image IDs and locations of the occurrences of j through the inverted files. For each retrieved word occurrence d in image i, we calculate the offset of d and j and increment the corresponding offset bin of image i.

$$S_{i,x_d-x_j,y_d-y_j} += 1 \tag{5.1}$$

   where, (x_d, y_d) and (x_j, y_j) are the x and y axis of the locations (in the quantized image space) of d and j, S_i are the scores for image i.
3. Calculate the number of co-occurring n^{th} order features for each image i by

traversing each bin m.

$$\widehat{S_i} = \sum_{m>=n} \binom{S_{i,m}}{n} \tag{5.2}$$

4. Obtain the final score S_i^* of each image by normalizing $\widehat{S_i}$ with its L2-norm.

5.2.3 IDF Weighting for HOFs

We further consider the *idf* weights of visual words. The idf value for a word w_j is calculated as $log(N_j/N)$, where N_j is the number of images that contain w_j and N is the total number of images. For the sake of efficiency, we define the idf weight of a HOF p as the summation of the idf weights of the words composing it: $idf(p) = \sum_{w_j \in p} idf(w_j)$. Thus, the algorithm for encoding idf weights is quite similar to coding the local features in Section 3.2.3. The inner product of the HOF histograms will take similar form as Equ. 3.7.

To calculate the similarity scores of all images, we modify the voting procedure in the previous section as follows. For each image i, other than the number of words $S_{i,m}$ in each offset bin m, we also keep the summation of the idf weights of these words $D_{i,m}$. At step 2, for each word j, we update both these values:

$$S_{i,x_d-x_j,y_d-y_j} += 1 \tag{5.3}$$

$$D_{i,x_d-x_j,y_d-y_j} += idf(j) \tag{5.4}$$

Equation 5.2 for calculating the score of image i is changed to:

$$\widehat{S_i} = \sum_{m>=n} D_{i,m} \binom{S_{i,m}-1}{n-1} \tag{5.5}$$

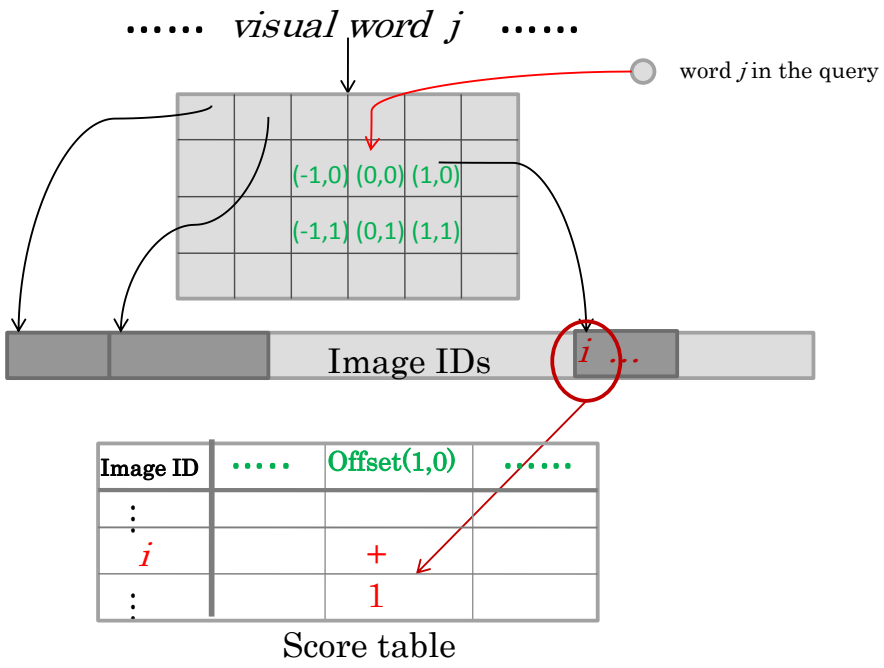


Figure 5.1: Inverted file structure and the illustration for updating the scores of images with this structure. Green numbers are the offsets of word j and the word occurrences in the database, which are calculated online using the location of j.

5.2.4 Index Structure

We discuss about the storage strategy of the inverted files. For storing the images that contain a visual word, there are two standard strategies for the inverted files [42, 94]. The first one is to keep one entry for each image, and store its ID and the term frequency of this word in this image. The second one is to keep one entry for each word occurrence, and avoid the storage for the term frequencies. With a large vocabulary, the memory usage is almost equivalent for the two strategies, since the same word rarely occurs multiple times in one image [42] [13].

To store the locations of the words, one simple way is to adopt the second

strategy and associate each word occurrence with the location in which it occurs. Since our algorithm uses only the quantized locations [1], an alternative storage method is illustrated in figure 5.1. For each visual word j and each quantized image location (x, y), we store the list of images that contain word j at location (x, y). This data structure only increases the memory by the pointers of the locations. Supposing we have 1M images with 2 billion features and the image space is quantized to 10 by 10 grids, the first method costs additional 2G bytes of memory to store the locations, while the second one only increases the memory by 100M bytes. Another advantage of this structure is that we do not need to calculate the offsets for each word occurrences in step 2. The offset can be calculated before referring to the image lists at a location (x, y) (Figure 5.1).

5.3 HOF with Min-Hashing

HOF can also be used to improve the retrieval accuracy of the min-hash method. The min-hash method [14, 16] is one popular dimension reduction technique that reduces the memory usage of inverted files and increases the searching efficiency, and is originally designed based on the bag-of-visual-words model. It is particularly suitable for near-duplicate image retrieval. We briefly review the min-hash algorithm based on BoW.

[1] Instead of quantizing the offset space as in Section 2.3, we quantize the image space to reduce the memory usage.

5.3.1 Min-hash Review

A number of independent random hash functions are generated. Each hash function f_j randomly assigns a real number to each visual word, and thus defines a permutation of the words. An image I is represented as a set A_I consisting of the words that occur in I. The min-hash value of the image A_I under function f_j is defined as the smallest word in A_I: $mf_j(A_I) = argmin_{w \in A_I} f_j(w)$. We call this mf_j a min-hash function, which has the property that the probability of two sets A_I and $A_{I'}$ having the same min-hash value is equal to their set overlap. The similarity of two images is defined as their set overlap.

$$p(mf_j(A_I) = mf_j(A_{I'})) = \frac{|A_I \cap A_{I'}|}{|A_I \cup A_{I'}|} = sim(A_I, A_{I'}) \tag{5.6}$$

If l is the number of times $mf_j(A_I) = mf_j(A_{I'})$ among the N min-hash functions we defined, the similarity of images A_I and $A_{I'}$ can be estimated as l/N.

For efficient retrieval, the min-hashes are grouped into k-tuples $F = (mf_1(A_I), mf_2(A_I), ..., mf_k(A_I))$ called sketches. The probability that two images A_I and $A_{I'}$ has identical k-tuples is $sim(A_I, A_{I'})^k$. The typical retrieval procedure estimates the similarity of the query with only images that have h identical sketches out of S randomly generated sketches.

5.3.2 Min-hash with HOF

To integrate the HOF to the min-hash algorithm, we first force one-to-one mapping from a min-hash function to a feature (a word occurrence) for each image. A min-hash function mf_j selects the word w in image I, which has the minimum value under function f_j. If the word occurs only once in the image, we set the

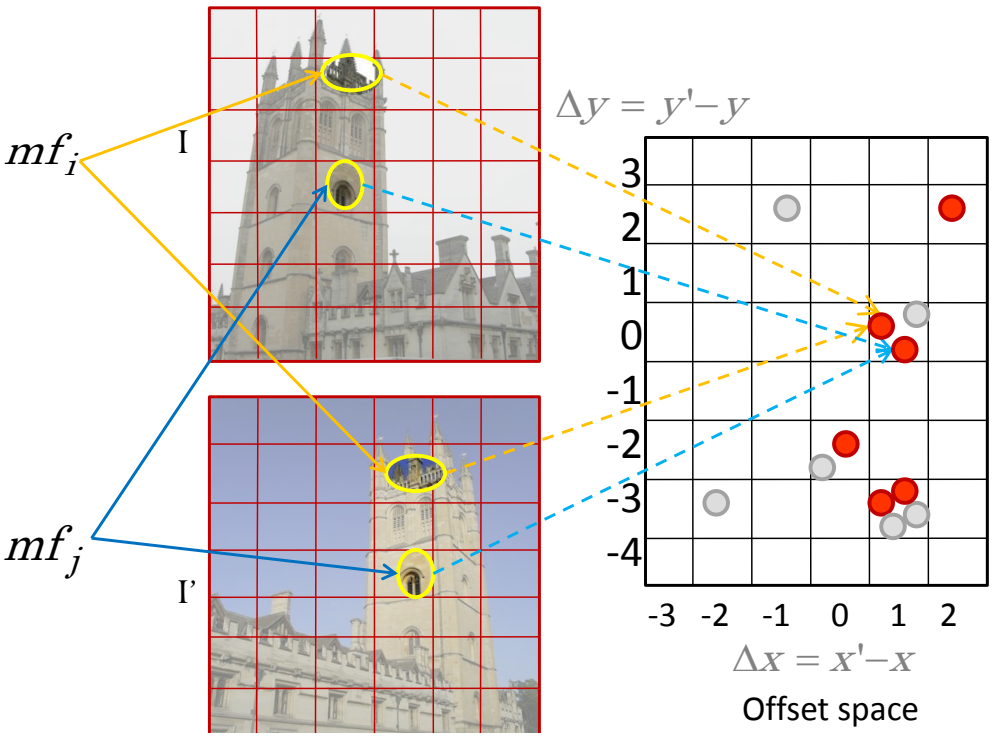


Figure 5.2: Illustration of the min-hash method with HOF.

function value $mf_j(I)$ to this occurrence. If the word occurs multiple times, we randomly select one of them and use the selected one as the function value. In practice, with a large vocabulary, most words has only one occurrence in the same image (more than 95% reported in [13]).

Thus, k min-hash functions will locate k features, a k^{th} order HOF, in an image. The similarity score $sim^k(I, I')$ of two images I and I' is defined as the probability that the HOF located by a set of k min-hash functions on these images are the same HOF. The calculation of this similarity is illustrated in figure 5.2. For each min-hash function mf_j, we locate the corresponding feature in each image. If the two features are the occurrences of the same word, that is, the min-hash value of the two images is the same, we calculate their offset and generate a vote on the offset space. k votes at the same bin on the offset space correspond to a

co-occurring HOF located by a set of k min-hash functions. Let m_s be the number of votes in bin s of the offset space and N be the total number of min-hash functions used; the similarity score $sim^k(I, I') = \sum_s \binom{m_s}{k} / \binom{N}{k}$. We use this similarity for ranking.

Collision probability

We look into the defined similarity score $sim^k(I, I')$, which is also the probability that a HOF collision occurs. This also equals to the probability that for k min-hash functions, 1) the min-hash values of all functions are the same for the two images; 2) the located feature pairs generate votes in the same bin on the offset space. Suppose the two images have $\rho^1(I, I')$ number of same word pairs. This also indicates that if we generate one vote for each same word pair (gray and red circles in figure 5.2), we have $\rho^1(I, I')$ total number of votes on the offset space. Because of its randomness, when a min-hash function has the same min-hash values for images I and I', it will randomly generate one vote among the $\rho^1(I, I')$ votes (red circles). Therefore, let M_s be the number of votes (gray and red circles) in offset bin s, and $sim(A_I, A_{I'})$ be the word set overlap defined in equation 5.6; the similarity score is:

$$sim^k(I, I') = sim(A_I, A_{I'})^k \frac{\sum_s M_s^k}{(\sum_s M_s)^k} \tag{5.7}$$

$$= sim(A_I, A_{I'})^k \frac{\rho^k(I, I')}{\rho^1(I, I')^k} \tag{5.8}$$

where $\rho^k(I, I')$ is the total number of co-occurring HOF of order k[2].

Note that the left part of equation 5.8 is the probability of a sketch collision. By using HOF, we further decrease the collision probability. Table 5.1 gives typical probabilities that a pair of relevant images has at least one sketch or HOF

[2]The HOF here include the phrases generated using the same word multiple times

method	S=512		S=1024	
	k=2	k=3	k=2	k=3
sketch	0.61	0.04	0.85	0.08
HOF	0.075	0.002	0.15	0.004

Table 5.1: The probabilities that at least one sketch or HOF collision for relevant image pairs among S number of sketches or HOF. k is length of the sketch or the order of HOF.

collision among S number of sketches or HOF[3]. Even for 2nd order, the probability is quite low for at least one HOF collision among 1024 HOFs. The proposed algorithm works because with N number of min-hash functions, we have already considered $\binom{N}{k}$ number of HOF. Therefore, when we have 512 min-hash functions, we can ensure to have at least one HOF collision with probability 1.0 for both $k = 2$ and $k = 3$.

5.4 Experiments

5.4.1 HOF with Inverted Index

Datasets: The experiments are conducted on three publicly available image retrieval datasets: *Oxford 5K Dataset* [94] and *Flicker 1M dataset* [42]. *Oxford 5K dataset* contains 5062 images with more than 16M features. It also provides 55 test queries of 11 Oxford landmarks with their ground truth retrieval results. *Flicker 1M dataset* contains 1M images with more than 2 billion features. This dataset is added as distractors to the Oxford dataset to test the scalability of our system. To ensure the compatibility of the experiments, we use the public

[3]The probability is calculated as the median of the probabilities of the relevant image pairs on the University of Kentucky dataset [85] with a 100K vocabulary

available descriptors (SIFT features on hessian affine regions or maximally stable extremal regions) on the dataset web pages. To create the vocabulary, we adopt the approximate k-means [94] whose code is published by the authors.

The retrieval accuracy is measured with the mean average precision (mAP) score generated as follows: the precision-recall curve is created for each query to calculate the average precision (AP). The mAP is defined as the mean of the APs of all queries.

Parameter effects: We first examine the effect of the main parameters in our approach on the Oxford 5K dataset. All the experiments here are conducted with 1M vocabulary size. Table 5.3 shows the mAP scores of different order features. We quantize the image space with 10 by 10 steps [4]. The BoW model (1st order) has a mAP score of 0.634, and using 2nd order feature improves the mAP score to 0.696. The searching algorithm with a longer HOF corresponds to a more rigorous geometry modeling. 2nd order is also the optimal order among the orders from 1 to 5. Since we only retrieve images with common HOF, a longer HOF may also leads to a lower recall when the shape deformation of the relevant images is large. Table 5.2 shows the effect of different number of steps for quantizing the image space. The same number of steps is used for x and y axis of the image space. We use 2nd order features in these experiments. A larger number of steps corresponds to a more rigorous spatial modeling. Increasing the number of steps also increases the memory usage for storing the accumulated scores in the offset bins. Therefore, we choose to use 10 steps with 2nd order features in our retrieval system.

[4]To lessen the problem of hard quantization of the image space, and we also quantize the offset space by a factor of 2, that is, we merge the neighboring offset bins. The left sides of Equation 5.3 and 5.4 will be changed to $S_{i,\llcorner(x_d-x_j)/2\lrcorner,\llcorner(y_d-y_j)/2\lrcorner}$ and $D_{i,\llcorner(x_d-x_j)/2\lrcorner,\llcorner(y_d-y_j)/2\lrcorner}$

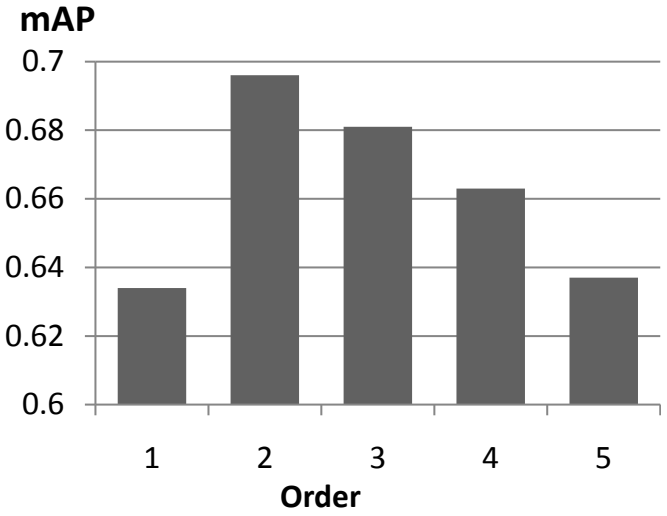


Figure 5.3: The effect of parameter changes on Oxford 5K dataset: mean average precisions with HOF of different orders. 1st order corresponds to the BoW model.

# steps	mAP	# r. Imgs
2	0.657	3949
5	0.682	3334
10	0.696	2597
15	0.698	1510
20	0.692	1354

Table 5.2: The effect of parameter changes on Oxford 5K dataset: the change of mAP and average number of retrieved images when changing the number of quantization steps of the image space.

Comparison: Table 5.3 compares the retrieval accuracy (mean average precision) of our approach with the other methods under different vocabulary sizes on the Oxford 5K dataset. The results showed that encoding spatial information (HOF) to the BoW model (BoW) can improve the retrieval accuracy. More significant improvement is made on smaller vocabulary sizes, because the visual words are more ambiguous. Searching with HOF also performs better

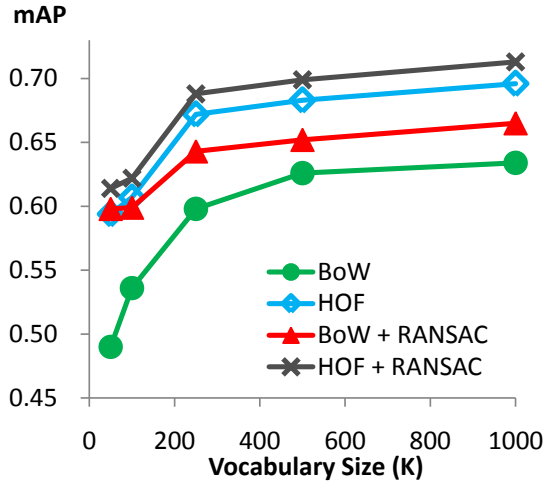
than the BoW model plus a RANSAC post-processing (BoW+RANSAC, for this method, we cite the results reported in [94]), especially on larger vocabularies. The reason is that our approach can provide spatial examination for the whole database, while the RANSAC only runs on the top images resulted by BoW and thus will be affected a lot when the precision of the top images is low. On the smaller dictionary, BoW+RANSAC performs better than HOF. The reason is that when the visual words are very ambiguous, the HOF will also be affected, while the RANSAC is less influenced since it is running on raw features. We further apply RANSAC on the top 400 images ranked with the HOF (HOF+RANSAC), which provides the best results among these methods. The improvement from HOF to HOF+RANSAC is resulted from the spatial information the HOF failed to capture, such as scale or affine transformation, or the errors in HOF caused by visual word quantization.

We also compare the proposed HOF with the Spatial Bag-of-Feature method (SBOF) [8], which also integrate spatial information into the BoW model. The SBOF uses the concatenated histogram of the histograms in local bins and introduces the transformation invariance by reordering the histogram so that it starts from the bin with maximum number of words. We encode more spatial information by considering the interaction between the local bins, and our method is more robust to invariance since we do not fix one-to-one matching between bins of two images. They report 0.651 mAP with the SBOF on the 1M vocabulary [8], while the proposed HOF achieved 0.696 mAP.

Analysis: Figure 5.4 presents the precision-recall curves of the BoW model and the proposed HOF for two example queries. These curves are also typical among the curves for other queries. Our approach with HOF improve the pre-

Vocab	BoW	HOF	BoW + RANSAC	HOF + RANSAC
50K	0.490	0.594	0.599	0.614
100K	0.536	0.607	0.597	0.622
250K	0.598	0.672	0.633	0.688
500K	0.626	0.683	0.642	0.699
1M	0.634	0.696	0.653	0.713

(a)

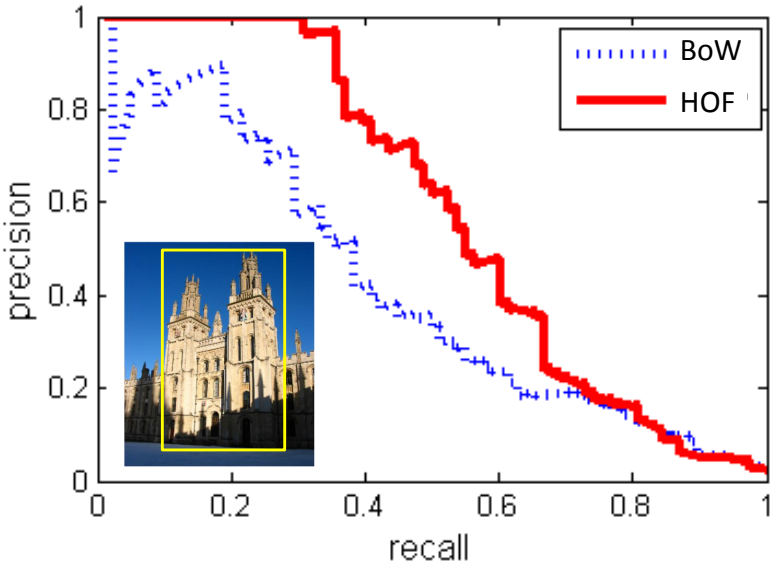


(b)

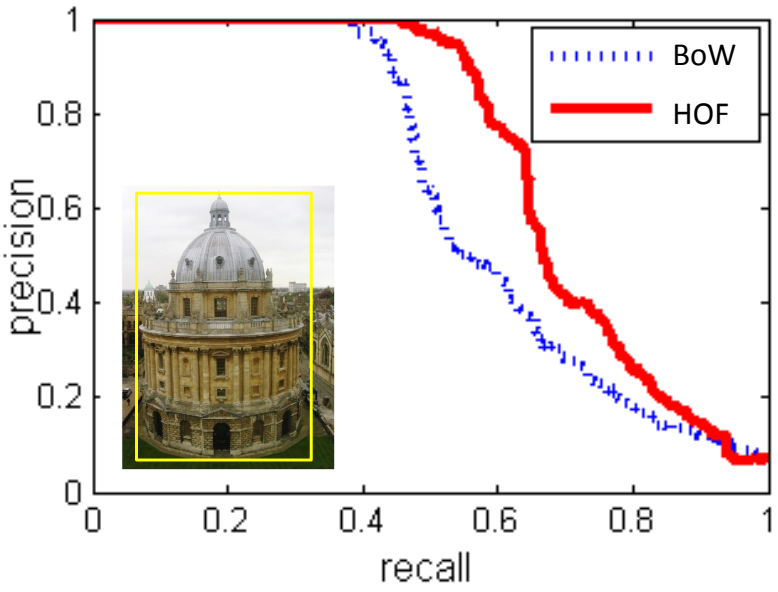
Table 5.3: Comparison of the performances of HOF and BoW with different vocabulary size for the Oxford 5K dataset.

cision of the BoW model at lower recalls. Close precision is shown when the recall reaches a certain point. We found the reason of this phenomenon is as follows. There are some relevant images which have few matched visual words with the queries. For these images, we can neither find co-occurring HOF with the queries. Therefore, our method with HOF fails to improve the ranks of these images.

Flicker 1M: We show the retrieval accuracy when adding the Flicker 1M images in figure 5.5. The proposed method with the HOF consistently improve



(a)



(b)

Figure 5.4: The precision-recall curve for example queries on the Oxford 5K dataset. (a) is for query *all_souls_1*, and (b) is for *radcliffe_camera_3*.

# Images	BoW	HOF
105K	0.509	0.604
205K	0.479	0.581
505K	0.443	0.554
1M+5K	0.413	0.532

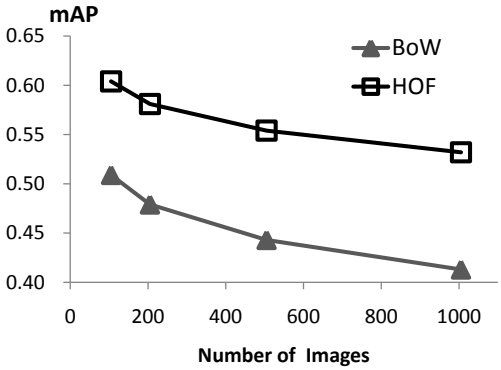


Figure 5.5: The mean average precision of Oxford 5K dataset combined with Flicker 1M images as distractors.

the ranking results of the BoW model on different number of images. The HOF outperforms the BoW by 12% on the 1M images.

Computational cost: We run our experiments on a *single CPU* of a 2.26G Quad-Core Intel Xeon server with 12G memory. The Flicker 1M + Oxford 5K dataset contains around 2G features and therefore, the size of the inverted files is around 8G. Compared with the BoW model, the HOF method needs additional memory to store the relative pointers of the locations as presented in figure 5.1 and the scores of offset bins. The additional memory required is around 500M, insignificant comparing to the inverted files. Table 5.4 summarizes the memory usage as well as the running time. Feature extraction time is not included. The proposed HOF achieves a significant improvement in retrieval accuracy with little speed penalty. In comparison, the RANSAC spatial verification on top 400 images takes more than 4 seconds per query. The increased runtime (around 0.1 seconds) is mainly because the scores of the offset bins cannot be saved to the Cache (8M), and the update of them requires random access to the memory.

Method	Memory	Runtime	
		Quantization	Search
BoW	8.1G	0.89s	0.137s
HOF	8.6G		0.248s

Table 5.4: The memory usage and average runtime per query on the Flicker 1M dataset.

5.4.2 HOF with Min-hash

We evaluate the proposed min-hash method with the HOF on the University of Kentucky dataset [85] which is also used in the previous min-hash paper [16]. We use the same experiment setting as in [16,85]. The retrieval accuracy is measured as the average number of relevant images in the top four retrieved images. We choose to integrate the HOF into the min-hash method with the similarity function using histogram intersection [16], since the method with histogram intersection reports the best results in [16]. We use sketches of length 2 in the same way as the traditional min-hash, and compute similarity scores with HOF for images with at least one sketch collision with the query image. We adopt the same number of sketches with which the best performance is achieved for each vocabulary size in [16]. We build a 100K vocabulary using the same features as [14].

Table 5.5 presents the average number of relevant images in the top 4 images for the min-hash method with BoW and HOF. The results using different number of min-hash functions are presented. By encoding the spatial information using the proposed HOF, our approach outperforms the min-hash method with BOW. The improvement is less significant when fewer min-hash functions are used, because the probability of a HOF collision is low in this case.

method	# of min-hash functions				
	268	512	640	768	896
BoW	2.88	3.07	3.12	3.14	3.17
HOF	2.96	3.14	3.20	3.24	3.26

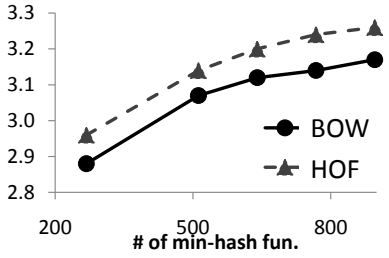


Table 5.5: The number of relevant images in top 4 images on the University of Kentucky dataset for the min-hash methods with BOW and HOF (2nd order).

5.4.3 Discussion

We discuss the limitations of our approach in this section. First, like most other image retrieval systems, our approach is built upon the visual word representation. As already discussed, if few matched visual words are found for the relevant images, our algorithm cannot correct them. This problem may be solved with a query expansion [15] on raw features in the post-processing step. Second, since we model the spatial interaction among the visual words, our current algorithm can not be applied to some dimension reduction methods that conduct linear transformations on the histogram of visual words [43, 44, 138]. However, as we have shown, as long as the dimension reduction method retains the word representation as the min-hash method [14], our approach is still applicable.

CHAPTER 6

HOFS FOR VIDEOS

In this Chapter, we further extend our HOF based approach from images to videos.

6.1 Introduction

We explore the video application, activity recognition, which has attracted increasing interest recently. An activity can be defined as a certain spatial and temporal pattern involving the movements of a single or multiple actors. The recognition task requires capturing enough spatial and temporal information to distinguish different activity categories, while handling the large intra-category variations. Also, most video analysis applications, such as surveillance, require high computational efficiency.

The local feature based approaches have become popular for activity recognition. Similar to local feature in images, a local feature in videos captures the local movement and appearance of a local region in a video, and thus can be ambiguous; For example, using the local features alone, the activities of "push" and "kick" can be quite similar when it is hard to tell whether a movement is from a person's hand or foot due to video resolution.

To better distinguish different types of activities, many works have been done to incorporate spatio-temporal relationships among the local features [21, 32, 40, 76, 86, 107, 121, 127]. Due to the computation limit, previous work usually only creates a combination from neighboring features in space and/or

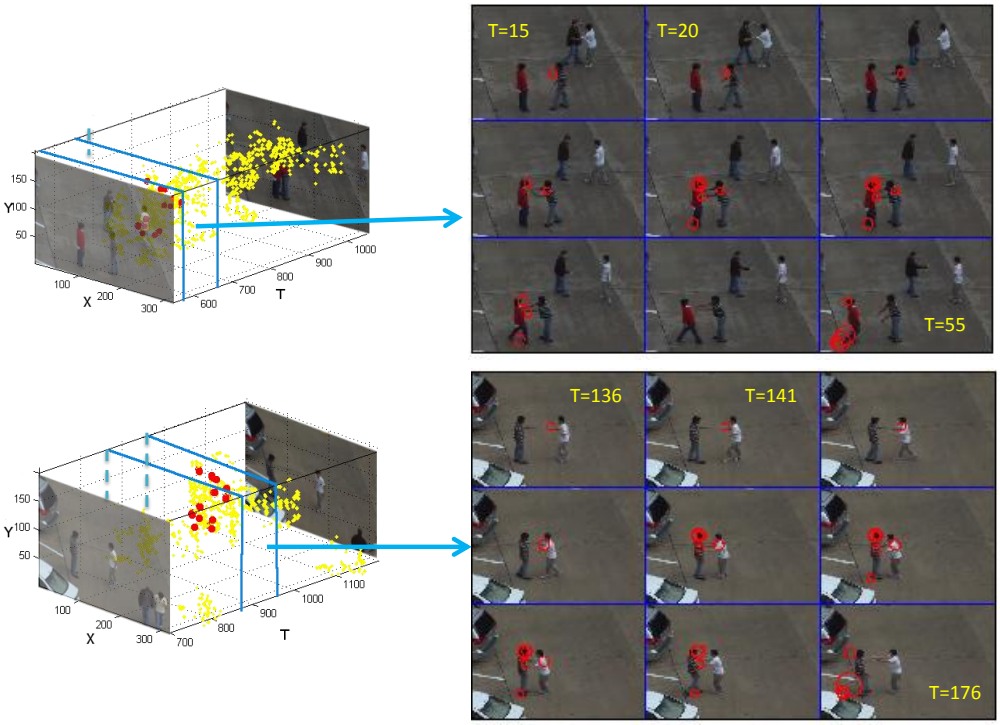


Figure 6.1: An example co-occurring spatio-temporal (ST) HOF of two video sequences. The left images show space-time locations of the local features in each video. Red points indicate the features composing the co-occurring phrase. The right images show the frames (sampled at rate 5, frame index is shown) composing the phrase. Red circles indicate the local features composing the phrase. The ST-HOF captures the causality relationships among body parts from different individuals of a long time span for the same activity "push".

time. We extend the proposed HOF to capture both spatial and temporal relationships of the local features, and call it "spatio-temporal HOF (ST-HOF)". A ST-HOF of order k is defined as a combination of k words in a certain spatial and temporal structure. Hence, it encodes rich temporal ordering and spatial geometry information of local words. Fig. 7.1 illustrates example ST-HOFs in two videos, both of which include the "push" activity at different time stamps. The ST-HOFs capture the patterns, e.g. the hand movement of one person in one frame and the foot movement of another person several frames later. The

illustrated ST-HOF consists of local movements that are not neighboring each other in both space and time. Moreover, the same phrase, which characterizes the "push" activity, occurs at different locations and time stamps in the two videos.

We extend the HOF algorithm introduced in Chapter 2 and Chapter 3 to the space-time domain. We also provide an algorithm that makes the ST-HOFs speed invariant to deal with different speeds or durations of the activities. In addition, we propose an algorithm that can update the ST-HOF kernel values incrementally when a new frame arrives, thereby enabling us to efficiently detect the activities from video streams in an online fashion.

6.2 Related Work

Holistic or body part model: Early works for activity recognition have focused on activities of a single person in controlled environments. Recognition is performed by modeling spatio-temporal patterns of the entire human body [24,79]. Holistic models can capture rich structural information of different body parts, but do not generalize well to handle intra-category variations. Hierarchical body representation [84] and body part based methods [1,48] can tolerant more intra-category activity variations, but still require robust body part tracking or segmentation.

Bag of words: Local features in the space-time domain have become increasingly popular. Local regions are extracted directly at interest points with high saliency [20, 61, 62] or by dense sampling at different locations in video frames [56, 122, 126]. The extracted regions are represented with descriptors re-

flecting the local motions or appearances [20,61,62,75], which are further quantized into the so called "visual words" [20,62,75] or "atomic actions" [122,136] with K-means or graphical models such as LDA and PLSA. The bag-of-words (BoW) model [20,62,119,126] represents a video with the histogram of the visual words, and has shown inspiring performance for activity recognition.

Spatio-temporal modeling: Many techniques have been proposed to incorporate spatial and temporal information into the BoW model. *Space-time pyramid matching* [62] captures absolute spatio-temporal locations of the local features with quantized space-time bins; therefore, it is not spatio-temporal shift invariant. Aligned space-time pyramid matching [21] relaxes the fixed bin matching, and identifies optimal matching between the bins of two videos. The method is shift invariant at the cost of discarding the spatio-temporal ordering among different bins. *Graphical Model based methods* [40, 49] captures the spatial and temporal dependencies by creating edges among the atomic actions in different frames in the LDA or MRF models. For efficient inference, the models only have edges for consecutive frames or neighboring regions, and thus do not capture long-range spatial and temporal interactions. *Trajectories of the local features* [80, 107, 128] are created by feature tracking methods and are capable of incorporating temporal information of the same feature in a certain period. However, exploring rich spatio-temporal relationships of different trajectories remains challenging. Messing et al. [80] use Markov chain to model the relationships of local features in a single trajectory, and treat an action as a bag of trajectories. Sun et al. [107] approximate the position of a trajectory as its center, and encode the neighboring relationships among different trajectories while neglecting the spatio-temporal ordering or long-range structures. *Hough Voting* or the *Implicit Shape Model* [127, 132] allow the local features to vote for the action

center in both space and time domains, and can encode space-time information of the words relative to the action center; however, the recognition process requires an iterative EM process to predict the center [127] or a preprocess for detecting the bounding box of the person [132]. *Mutual word relationships* have been explored [32, 34, 53, 76, 102]. Due to the high computational cost, previous works do not capture higher order and long range dependencies. Our ST-HOF based approach improves previous works with more spatial and temporal information among the local features yet less computation complexity, while providing discriminative learning at the same time.

6.3 Spatio-Temporal HOF

In activity recognition, a video can be represented as a collection of visual words in the *xyt*-space, which are created by clustering the local descriptors detected at local space-time volumes. Various promising local feature detectors, descriptors and clustering methods [20, 61, 75, 122, 126] can be adopted. Specifically, a video is denoted as $V = (w_i, x_i, y_i, t_i)$, where w_i is the word entry for feature i, and $(x_i, y_i), t_i$ are the space and time index of the feature respectively.

6.3.1 3D Correspondence Transform

A co-occurring ST-HOF of order k in two videos must have the same k words and the same space-time layout, as shown in Fig. 6.1. The algorithm for identifying co-occurring ST-HOFs is an extension of the correspondence transform algorithm proposed in Chapter 2 for 2D images. As illustrated in Fig. 6.2, for

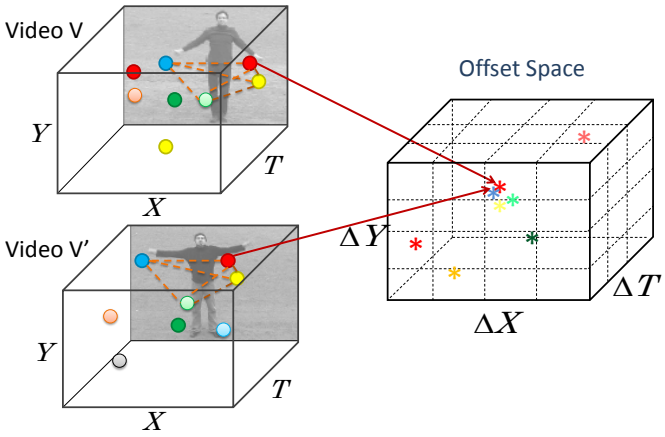


Figure 6.2: 3D correspondence transform. Circles represent local features, and colors represent word assignments. A co-occurring ST-HOF of order 4 is shown.

each pair of local features (w_i, x_i, y_i, t_i) and (w'_i, x'_i, y'_i, t'_i) that has the same word assignment $(w_i = w'_i)$ in the two videos V and V', we calculate their space-time offset as follows:

$$(\Delta x_i, \Delta y_i, \Delta t_i) = (x_i - x'_i, y_i - y'_i, t_i - t'_i),$$

and create a vote in the XYT offset space $(\Delta X, \Delta Y, \Delta T)$.

In the quantized XYT offset space, if we have k votes at the same location, we have a co-occurring ST-HOF of order k in the two videos with the same word entries and the same spatio-temporal structure. Thus, to count the number of co-occurring ST-HOFs of length k, we can simply use the XYT offset space similarly as the 2D case.

Similar as HOF for object recognition in Chapter 3, we use the inner-product of the ST-HOF histograms as a kernel for the SVM to classify a video to an activity category.

Activity Speed Variations: the same activity category can occur with different speeds and time durations in different videos. For example, people may

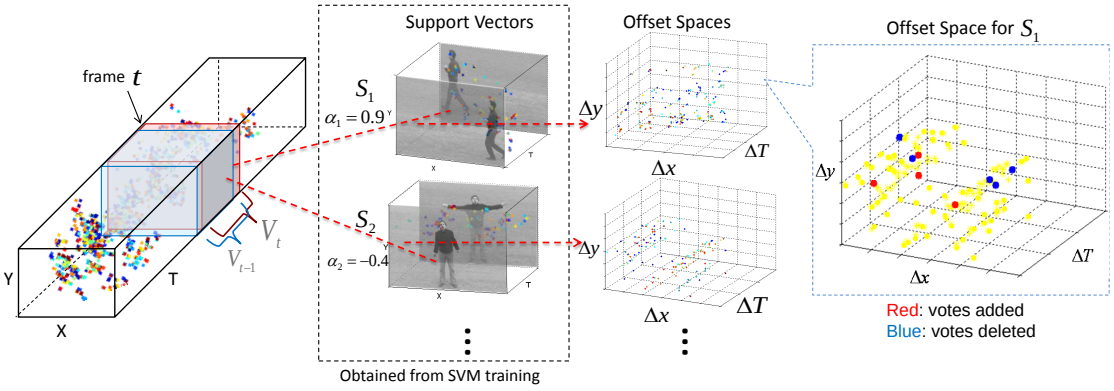


Figure 6.3: Illustration of the incremental kernel computing algorithm. The support vectors (selected training videos) S and their coefficients α (second column) are obtained during training, while the offset spaces for each support vector (third column) need to be computed online during testing. The right most image shows the enlarged offset space between the video segment V_t and support vector S_1.

"run" at different speeds, or take different time to form a group in order to "fight". To explicitly deal with this problem, we would like to detect k words from two videos as the same ST-HOF if their only structural difference is the temporal rates. To this end, we add another dimension to the offset space, which indicates the temporal scaling Δd, and allow each pair of words vote for multiple Δd. The offset location of a pair of features with the same word assignment is: $(\Delta x_i, \Delta y_i, \Delta t_i, \Delta d) = (x_i - x'_i, y_i - y'_i, t_i - t'_i \times \Delta d, \Delta d)$.

In this 4D space, if we have k votes at the same location, we have a co-occurring ST-HOF with a particular temporal scale difference Δd. In the experiments, we set the scale Δd to $1, 1.5, 1.5^2, .., 5$, which tolerates up to 5 times speed difference between activities of two videos.

6.3.2 Incremental Kernel Computing for Activity Localization

Some applications, such as surveillance or security monitoring, require online activity detection for the video streams instead of recognizing a temporally segmented video clips. One simple yet commonly used approach is to make predictions for every frame based on the video segment composed of the current frame and the previous T frames as the context. If some detection delay is allowed, the following T' frames are also included in the segment.

As illustrated in Fig. 6.3, to determine the activity category of the current frame, we need to compute the kernel values of the corresponding video segment with the support vectors. The essential for obtaining the kernel value of a video segment V_t and a support vector S_j is to compute the offset space which is created with the votes from the local features. As discussed in Section 6.3.1, this computational complexity is linear to the number of local features in the segment.

However, note that the video segment of the current frame and that of the previous frame have a significant overlap; we can further accelerate the kernel computation with an incremental algorithm. The difference between the video segments of the current and the previous frame only involves two frames, i.e., the current frame t, and the frame at $t - T$ (Fig. 6.3). If we have stored the offset spaces for the previous frame segment V_{t-1}, to compute the new offset space for the current V_t and a support vector S_j, we only need to add the votes made by the local features of the current frame, and delete the votes that were contributed by the features of the frame $t - T$.

When we add a vote at location l in the offset space that originally had m_l

Algorithm 1: Compute kernels for frame t and a support vector S_j
1. Initialize $K_k(V_t, S_j) = K_k(V_{t-1}, S_j)$
2. For each same word pair in video S_j and frame t of the test video (a) Compute their offset l, suppose the original votes at l is m_l (b) Increase $K_k(V_t, S_j)+ = \binom{m_l}{k-1}$ (c) Add one vote to location l Save the list of offset locations contributed by frame t
3. For each saved offset location l of frame $t - T$ (a) Decrease $K_k(V_t, S_j)- = \binom{m_{l-1}}{k-1}$ (a) Delete one vote from location l

votes, the number of co-occurring order-k ST-HOFs would be changes as follows:

$$K_k^{new}(V, V') = K_k^{old}(V, V') - \binom{m_l}{k} + \binom{m_l + 1}{k} = K_k^{old}(V, V') + \binom{m_l}{k-1}. \tag{6.1}$$

The change for the number of co-occurring order-k ST-HOFs when we delete a vote can be calculated similarly:

$$K_k^{new}(V, V') = K_k^{old}(V, V') - \binom{m_l - 1}{k-1}. \tag{6.2}$$

The algorithm for updating the kernel values is listed in Algorithm 1. Consequently, to compute the kernel values for the current frame and a support vector, we only need to perform the operations of Eqn.(6.1) and (6.2) $2N_t$ times, with N_t being the number of local features at frame t. This acceleration enables us to consider the long-term context (a large number of previous frames) without increasing the recognition time for each frame. This property is especially useful when detecting complex activities that usually last a long period of time.

6.4 Experiments

We first perform experiments on single person activities with the KTH dataset [103], YouTube Action dataset [75], and a hospital surveillance dataset. The KTH dataset is created in a controlled environment, while the hospital dataset is collected from real surveillance cameras in more complex environment and the YouTube Action dataset includes more pose and scale activity variations taken with shaky cameras.

Then we evaluate our performance for the multiple-person activity problem, which is an up-coming topic that recent work are addressing [7, 10, 32] with the UT interaction dataset [100] and the MPR dataset [10]. On the MPR dataset, we evaluate our incremental algorithm (Section 6.3.2) for online activity detection.

Baseline: We aim to verify that the proposed bag-of-ST-HOF representation outperforms BoW with exactly the same local features, same visual words, and same classifier (SVM). For BoW, we use the χ^2-kernel, which already captures the co-occurrence statistics of different words in a video. Therefore, the comparison of BoW and ST-HOF verifies whether the spatial-temporal layout of the words helps activity recognition. Moreover, we compare favorably with other state-of-the-art approaches [62,101,102,117,132] that incorporate spatial and/or temporal information to the BoW representation.

Method	500	1000	2000	4000
BoW	90.8	91.2	91.3	91.5
ST-HOF	**94.0**	**93.8**	**94.1**	**94.0**

Setting	BoW	ST-HOF	[62]	[132]	[102]	[101]
16 train / 9 test	91.5	**94.0**	91.8	n/a	n/a	91.1
5-fold	92.9	**94.6**	n/a	92.0	n/a	n/a
leave-one-out	91.9	**95.5**	n/a	n/a	94.2	93.8

Table 6.1: (a) Classification accuracies (%) with different vocabulary sizes on the KTH dataset. (b) Accuracies (%) under each train and test setting on the KTH dataset. We compare ST-HOF with other methods that incorporate spatio-temporal relationships to BoW.

6.4.1 Single Person Activity: KTH Dataset

We use the same local features as [62]. Features are computed with the published code by the authors [1].

Vocabulary Size: Table 6.1(a) compares ST-HOF with BoW using different vocabulary sizes under the same 16 train / 9 test settings as [62]. The performance of BoW decreases when a smaller vocabulary size is used since local words are more ambiguous. Meanwhile, ST-HOF achieves similar accuracies even with a smaller vocabulary, since the ST-HOFs capture the spatio-temporal relationships among the words, thereby increasing the discrimination.

Comparison: Table 6.1(b) compares ST-HOF with other methods that encode spatial relationships into BoW. By capturing higher order and more detailed spatio-temporal information with the ST-HOFs, our approach outperforms other methods: spatio-temporal pyramid matching [62], Hough voting [132], correlogram [102], and predefined spatio-temporal relationship rules [101].

[1] For fair comparison, feature HOF uses version 1.1 code, same as [62].

6.4.2 Single Person Activity: Hospital Surveillance Dataset

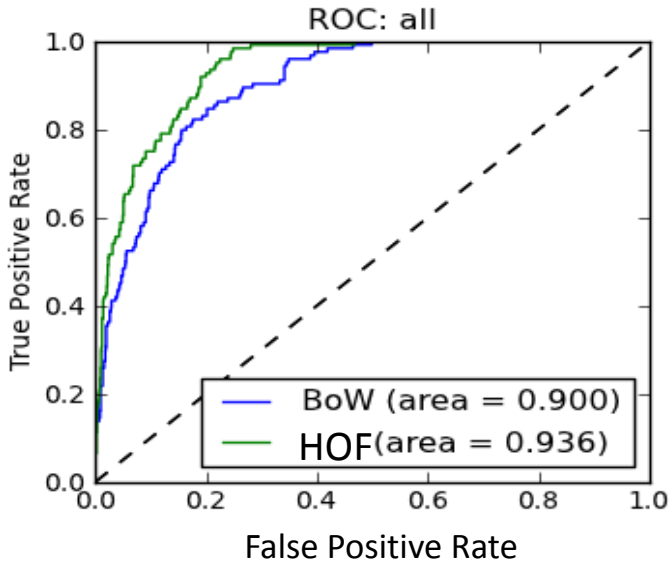
This dataset includes the realistic surveillance videos of 6 patients in the private sickrooms of a hospital[2]. The goal is to detect abnormal behaviors of the patients. The environment of this dataset is much more complex than the KTH dataset, since the patients conduct many variations of activities and many noise motion features are caused by non-person objects or other people, such as activities that a nurse clean the bed. To collect the ground-truth, we segmented the videos into 10 second clips, and manually labeled each clip. In total, the dataset has 124 positive (abnormal) examples, and 1067 negative examples. We perform a leave-one-patient-out experiment. We extract similar features as the KTH dataset, and create a vocabulary of 500. Figure 6.4 shows the performance comparing BoW and ST-HOF. For each individual patient (b) and all patients (a), our approach outperform the BoW method.

6.4.3 Single Person Activity: YouTube Action Dataset

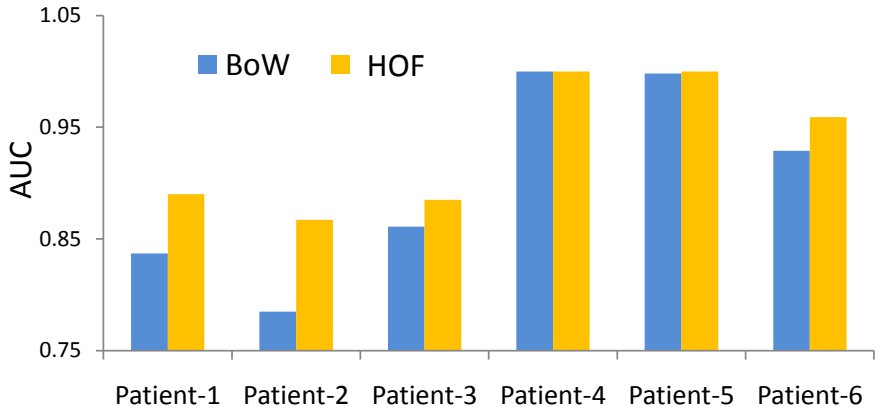
The YouTube action dataset [75] consists of 11 categories with 1168 videos. The videos are separated into 25 groups, which are taken in different environments or by different photographers. Following [75], we perform a leave-one-group-out experiment. We extract the local features with the code [62], and create a vocabulary of size 2000 with K-means.

Figure 6.5 shows the classification performance. The BoW achieves 63.7% accuracy averaged over the 11 categories. Our implementation of BoW achieves similar performance as the one in [75] (65.4%) when similar (motion) features

[2]For patient privacy, we cannot show the example images from the dataset.



(a)



(b)

Figure 6.4: Hospital Surveillance Dataset. (a) ROC curve for all patients. (b) The AUC score for each patient.

are used. With ST-HOF, we improve the performance by 9.2%. Our ST-HOF result (72.9%) is also better than the final result of [75] (71.2%), where additional static features, feature pruning techniques, and semantic vocabulary learning are adopted. These techniques that improve the local features are complimentary to our work. Therefore, further improvement can be expected when these techniques are applied. We notice the main improvement is the discrimination

of different *swing* activities, and different categories that both involve *jump* actions, such as *basketball shooting*, *trampoline jumping*, and *horse riding*. The main reason is that ST-HOF can capture the ST relationships among the local movements.

6.4.4 Multiple-Person Activities

We evaluate our approach using the UT interaction dataset [100], which consists of six activities of two-person interactions: *shake hands*, *hug*, *kick*, *point*, *punch*, and *push*. There are two sets in the dataset. Set 1 was recorded with relatively stable camera, while Set 2 is taken on a lawn in a windy day. Background is moving slightly (e.g. tree moves), and they contain more camera jitters. For each set, every type of activity has 10 video clips. These high-level activities are more complex, and involve a combination of atomic movements of two people over a period of time. Therefore, the BoW model which only captures the atomic movements may work poorly. This dataset was used in the activity recognition contest at ICPR 2010 [100]. We use the cuboid [20] as local features and a vocabulary of size 500.

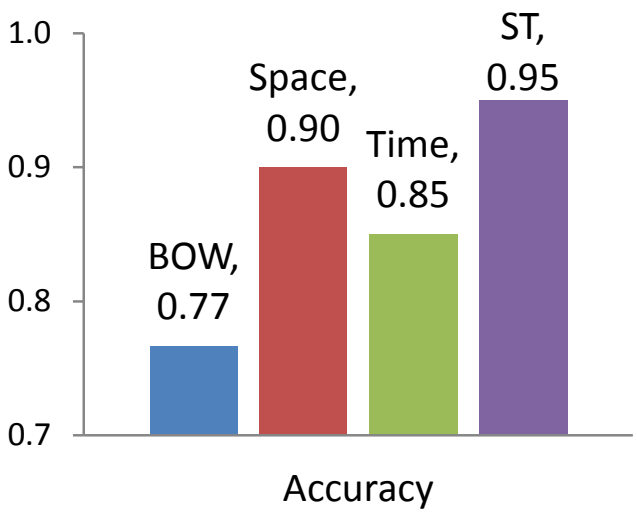
Comparison: We compare ST-HOF with BoW and the best results reported in the contest [117]. We adopt the classification settings described for the contest, where a 10-fold leave-one-out cross validation is performed. Figure 6.6 shows the confusion matrix for different methods on Set 1. Our implementation for BoW with a χ^2 kernel for SVM demonstrated 76.7% accuracy, which is similar to rates of the BoW method reported in the contest. Table 6.2 (b) compares the performance on both Set 1 and Set 2. By modeling the atomic moves

to the activity center, the Hough voting based method [117] improves BoW by around 7%. Using ST-HOF, we further outperform the Hough voting method by around 10%.

Analysis: Since the activities in this dataset involve more atomic movement interactions of different body parts and from different persons, the rich spatio-temporal interactions modeled with ST-HOF are beneficial. As shown in Fig. 6.6, BoW confuses *push* vs. *punch*, and *shake hands* vs. *hug*, since the local movements of each body part are similar for these activities. With the ST-HOFs, we can capture the spatio-temporal combinations, thereby better discriminating these activities. Although the Hough-voting based method captures spatio-temporal information of the words, the main disadvantage is that it does not support discriminative learning. Since the activities, such as *shake hands* and *hug*, still share some local movements, a generative learning cannot separate them well, especially when the spatio-temporal information of the local movements are modeled independently relative to the ST center in the Hough voting method. By formulating the ST-HOFs as a SVM kernel, we can perform discriminative learning with mutually relationships among the local words, and automatically learn the most discriminative ST-HOFs.

Analysis for Spatial and Temporal Information: We show the benefit of simultaneous spatial and temporal modeling with ST-phrases in Table 6.2(a). To incorporate space alone, we still use the HOF algorithms we proposed, but discarding the temporal domain. In other words, the temporal domain is modeled with BoW. To incorporate time alone, we ignore the space domain. According to the table, both spatial and temporal information improves BoW, and incorporating spatial information helps more than incorporating temporal information.

(a)



(b)

Dataset	Method	Avg.	Shake	Hug	Kick	Point	Push	Punch
Set 1	BoW	0.77	0.70	0.80	0.90	**1.00**	0.50	0.70
	Hough Voting	0.83	0.50	1.00	1.00	1.00	0.70	0.80
	ST-HOF	**0.95**	**1.00**	**1.00**	**1.00**	0.90	**0.90**	**0.90**
Set2	BoW	0.73	0.70	0.70	0.80	0.80	0.70	0.70
	Hough Voting	0.80	0.70	0.90	1.00	**1.00**	0.80	0.40
	ST-HOF	**0.90**	**0.80**	**1.00**	**1.00**	0.80	**0.90**	**0.90**

Table 6.2: Recognition accuracies on UT interact dataset. (a) The performance of incorporating spatial, temporal, and spatio-temporal information into BoW on Set 1. Note that ST-HOF is not a simple combination of spatial and temporal phrases. (b) The accuracies of different methods on both Set 1 and Set 2 of the dataset. For Hough Voting, we cite the reported results in [117].

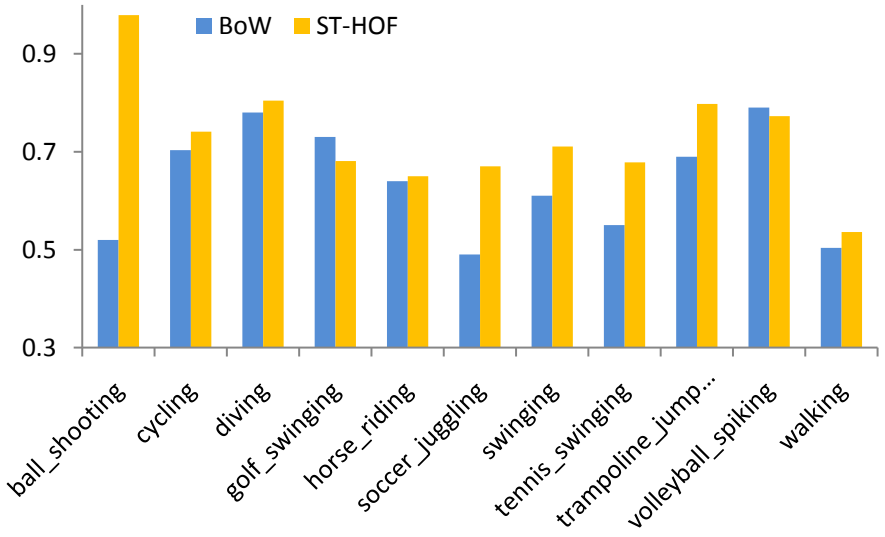
Simultaneous spatial and temporal modeling obtain the best result.

6.4.5 Online Group-Level Activity Detection

Finally, we evaluate on group-level activities with the Mock Prison Riot (MPR) dataset [10]. The dataset has 19 surveillance videos captured in an abandoned prison yard. Several volunteer correctional officers enact typical behaviors of the inmates, including 6 group-level events of interest: *group formation*, *group dispersion*, *group following*, *group chasing*, *group flanking*, and *group fighting*. The goal is to detect these events of interest from other random behaviors of the group of people. The duration of each event ranges from 2 to 30 seconds. These properties of the dataset require the system to use a long duration context when detecting events at each frame, and thus the speed of the detection algorithm is essential. Figure 6.7 gives sample snapshots of the dataset. We extract the same features as [139], which use a BoW approach on these features.

Online Detection with Incremental Kernel Computing: We perform event detection on continuous videos. For this task, we use the incremental algorithm proposed in Section 6.3.2. We randomly select 60% out of the 19 videos for training and the rest for testing. Prediction for every frame is made using observations from a four-second temporal window ($[t - 4s, t]$), i.e., the previous 4 seconds. Figure 6.8(a) shows the predicted probabilities for one of the test videos. Although BoW can detect the events of interest well, it generates much more false positives than ST-HOF because of the lack of discrimination of the events of interest with normal behaviors. Figure 6.8(b) compares ST-HOF with previous works using the AUC (area under curve) scores of ROC curves for different categories. ST-HOF improves the previous works, especially for *group forming*, *group following*, and *group fighting* events, where the local group changes are quite confusing with those of random behaviors.

Running Time: We implement our approach with C++ and Python on an Intel 2.4G dual-core computer. Excluding person tracking and feature extraction (around 0.02s per frame), classifying a 640× 480 frame using a 4-second context takes less than 5 millisecond with the incremental algorithm proposed in Section 6.3.2. Therefore, we can perform event detection in real time for continuous video streams. In comparison, directly classifying its 4-second context for each frame takes more than 200 millisecond.

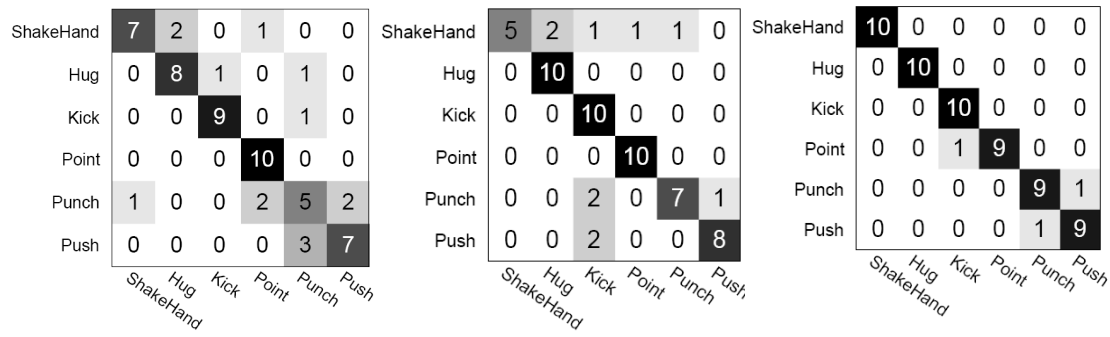


(a)

b_shooting	.98	.00	.00	.00	.00	.00	.00	.02	.00	.00	.00
cycling	.01	.74	.01	.00	.07	.00	.03	.02	.01	.01	.08
diving	.00	.04	.80	.00	.02	.02	.02	.03	.02	.02	.02
g_swinging	.03	.00	.04	.68	.00	.08	.01	.10	.02	.00	.03
h_riding	.00	.13	.02	.00	.65	.02	.00	.05	.01	.00	.12
s_juggling	.00	.01	.01	.09	.00	.67	.02	.07	.09	.00	.03
swinging	.00	.03	.02	.02	.01	.02	.71	.00	.08	.03	.08
t_swinging	.02	.00	.02	.06	.02	.07	.04	.68	.00	.06	.02
t_jumping	.00	.01	.00	.00	.06	.06	.06	.01	.80	.00	.01
v_spiking	.02	.02	.03	.02	.00	.00	.05	.06	.00	.77	.04
walking	.00	.11	.04	.01	.11	.10	.01	.03	.04	.02	.54

(b) average: 72.9%

Figure 6.5: (a) Accuracies for each category of the YouTube action dataset. Average accuracies for BoW and ST-HOF for 63.7% and 72.9%, respectively. (b) Confusion matrix using ST-HOF.

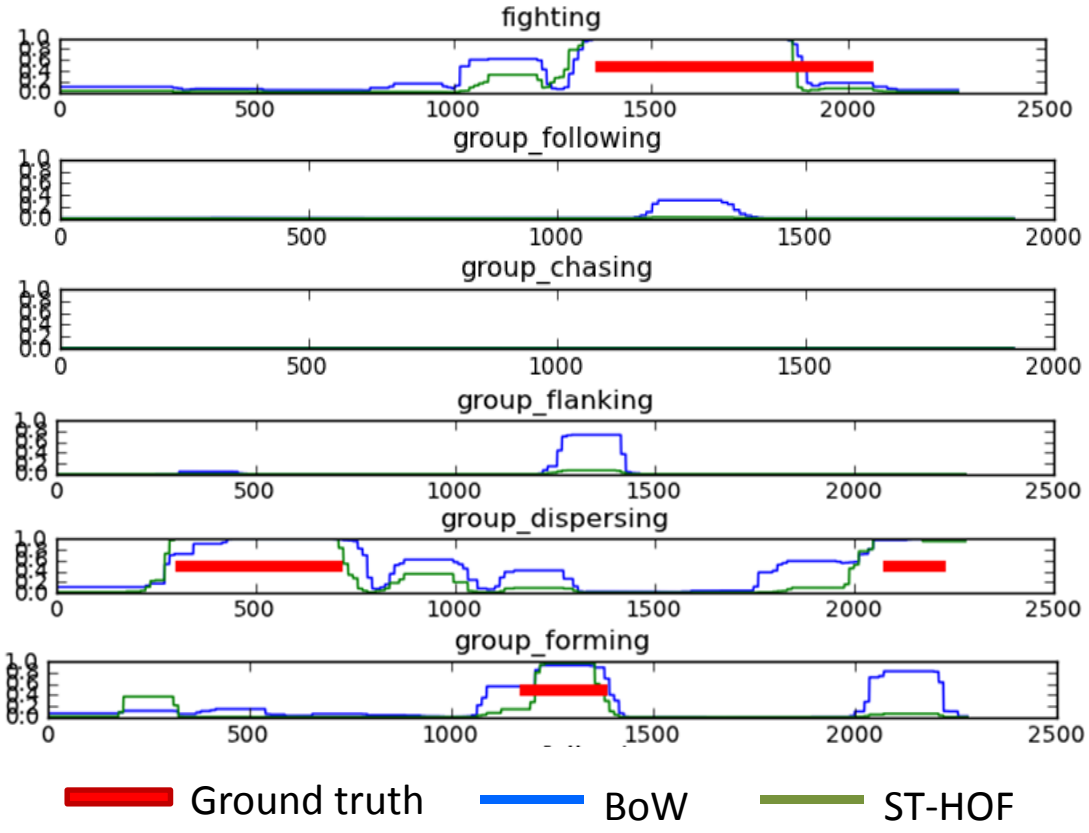


(a) BoW (b) Hough Voting [117] (c) ST-HOF

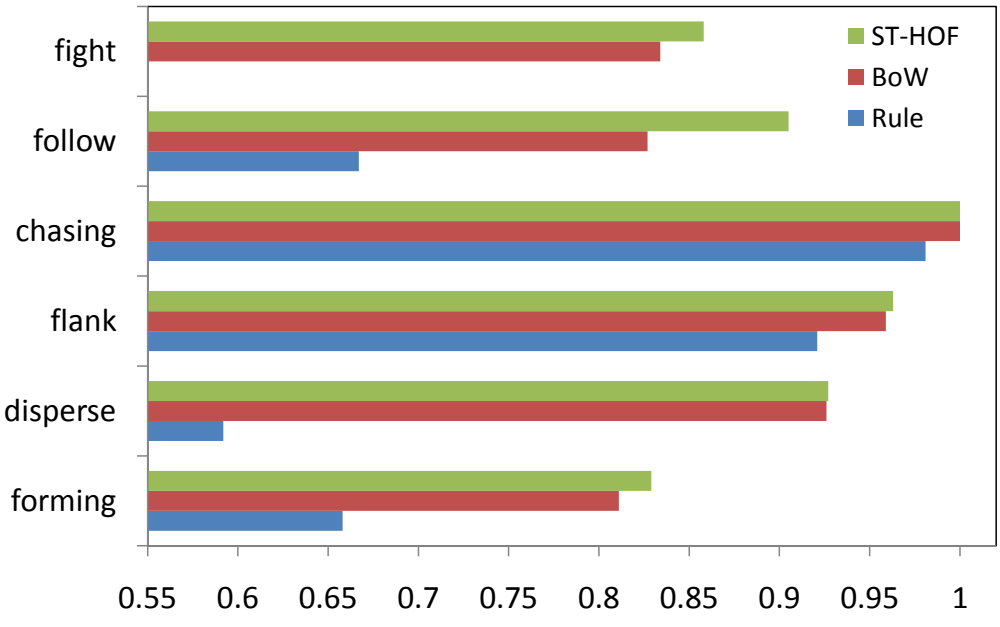
Figure 6.6: Confusion Matrices on the UT interaction dataset (Set 1).

(a) group dispersion (b) group formation (c) approaching (d) group flanking (e) group fighting

Figure 6.7: Example scenarios we aim to recognize from videos of the MPR group dataset.



(a)



(b)

Figure 6.8: (a) The predicted probabilities (vertical) of various events at each frame index (horizontal) for a test video. (b) Comparison of area under curve (AUC) scores for each event category on the whole test dataset. We compare the rule based method [10], BoW [139], and ST-HOF. Note that the rules for *fighting* are not defined in [10].

CHAPTER 7

PIXEL-LEVEL LABELING: FULLY-CONNECTED CRFS

In this chapter, we explore the spatial modeling for another vision problem: pixel-level labeling. Unlike the problem in previous chapters, where the goal is to obtain an estimation for the entire image or video (either a label or a similar image of the input), the task of this chapter makes a prediction for every pixel in an image. One of the most popular application would be object-based image segmentation, which segments an image by classifying every pixel of the image into an object category.

The Conditional Random Field (CRF) is probably the most popular tool for this task. Nodes of the CRFs capture the local appearances, while the edges make spatial connections of the local labeling. CRFs used in practice typically have edges only between adjacent image pixels. To represent object relationship statistics beyond adjacent pixels, prior work either represents only weak spatial information using the segmented regions, or encodes only global object co-occurrences.

We propose a unified model that augments the pixel-wise CRFs to capture object spatial relationships. To this end, we use a fully connected CRF, which has an edge for each pair of pixels. The edge potentials are defined to capture the spatial information and preserve the object boundaries at the same time. Traditional inference methods, such as belief propagation and graph cuts, are impractical in such a case where billions of edges are defined. Under only one assumption that the spatial relationships among different objects only depend on their relative positions (spatially stationary), we develop an efficient inference algorithm that converges in a few seconds on a standard resolution image,

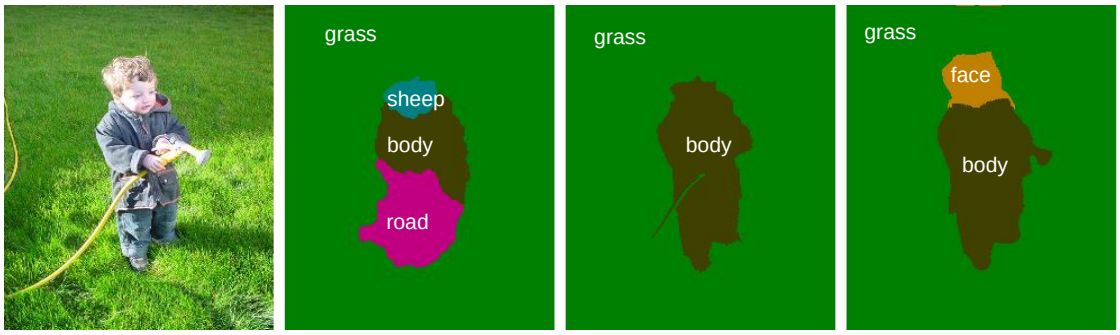
where belief propagation takes more than one hour for a single iteration.

7.1 Introduction

Object-based image segmentation is one of the most challenging problems in computer vision. Given an image, the goal is to label every pixel with a predefined object category. A good algorithm for this task should be able to incorporate as much context as possible to ensure accurate object categorization, while providing precise segmentation along the object contours. Computational efficiency is also important, especially for high resolution images.

The most popular method is based on the conditional random fields (CRFs) defined over the image pixels, which was originally proposed in the TextonBoost work [105]. Unary potentials in the CRFs capture the low level cues with local texture, color, and location [50, 104, 105]. Edge potentials are typically defined for 4 neighbor pixels to smooth out the prediction. Although this 4-neighbor grid CRF has shown promising results for object segmentation, it fails to capture long-range context information.

Many techniques have been proposed incorporating longer-range information and more contexts by augmenting the traditional CRF with additional potentials, including global image features [112], top-down object detection results [3, 35, 69, 124, 131], label consistency in the same segmented regions [31, 41, 57, 90, 97, 99], and the global co-occurrence statistics among object categories [55, 58]. Efficient inference algorithms have also been associated with the augmented CRFs.



(a) Image (b) Grid CRF (c) P^n+Co (d) Ours

Figure 7.1: Comparison of the object-based image segmentation results using our methods and previous works. (b) 4-neighbor grid CRF [105] (c) Robust P^n model plus object co-occurrence statistics [58] (d) Our method with a fully connected CRF, which captures both long-range color contrasts and the object spatial relationships in addition to their global co-occurrences. Our method preserves the object contours without pre-segmentation and is able to place the *face* at the right place even if its unary probabilities are small. The result with our method is obtained in less than 3 seconds.

In this paper, we further incorporate the CRFs with pixel-wise spatial relationships among objects, in addition to their co-occurrences in the entire image. We model the object segmentation problem with a fully connected CRF, which allows every pair of pixels in an image to connect with each other. Unlike the grid CRFs, where the edges only serve for the contrast sensitive smoothness, the proposed CRF has edge potentials that encode both color contrasts and spatial arrangements of different object categories.

The context of spatial object interactions has been explored by many previous works on object detection [19, 109], and has been proved of its effectiveness. However, in terms of object segmentation, encoding this information at pixel level into the CRFs remains challenging due to the computational cost. Previous methods [31, 90, 112] first segment the image and then model the ob-

ject relationships over the regions. Working on the regions largely reduces the computational complexity and makes it tractable to capture the semantic interactions between every pair of regions. However, the main disadvantage of the segment based methods is the errors introduced by the initial segmented regions. As discussed in [50, 57], this may result in dramatic errors in the final object labeling. Multiple segmentations [99] ease the problem, but increase the computation and may still leave some errors. As shown in [50, 55, 57], directly augmenting the pixel-wise CRFs usually works better. Moreover, the irregular segmented regions usually capture only weak geometry, such as the co-occurrence in the entire image [97, 112], four binary relationship indicators ("inside", "outside", "above", "below") [31, 55], or locations defined over the 3×3 grid of an image [90]. We are able to capture more detailed spatial information with the pixels and preserve object contours at the same time.

A fully connected CRF over image pixels has N^2 number of edges, where N is the number of pixels in an image. For an image of a resolution of 213×320, the number of edges is more than 5×10^9. The current inference methods [51, 52, 98, 123, 134] are impractical in such a case. We propose an efficient inference algorithm for the fully connected CRF, which reduces the complexity at every iteration to $O(N \log N)$. The proposed algorithm relies on one constraint on the edge potentials: Given two object categories, the spatial potentials over two pixels depend only on their relative positions. That is, we assume that the likelihood that two categories co-occur at two particular pixels is only determined with the offset of these pixels. This stationary assumption is a standard assumption, and was made by almost all previous context modeling works [19, 31, 55, 90, 109]. Under this assumption, we show that the gradient of the Quadratic Programming (QP) relaxation of the fully connected CRF can be

efficiently computed with the help of convolution. QP relaxation was proposed recently in [98] as another inference method for CRFs and has been proven to be a tight relaxation and solve exactly the MAP problem of the original CRFs. With the proposed algorithm, the QP optimization converges in a few seconds, while max-product belief propagation and the original QP relaxation take more than one hour even for a single iteration on the fully connected CRF.

Efficient inference for fully connected CRFs has also been explored by a simultaneous and independent work of Krahenbuhl and Koltun [54]. The main difference is that they make the Gaussian assumption of the *stationary* edge weights and use a Potts model with only label consistency for similar colors. We relax the assumption to any distribution for the stationary weights. With this more general assumption, we can encode more general statistics in the edge weights. For example, we can model the relative spatial relationship (not only *distances*) among different categories, such as the geometry relationship between cow and grass. We also provide a different inference algorithm for this more general problem.

The main contributions are as follows: 1) we present a unified model that augments the traditional CRFs to capture the object spatial relationships with fully connected edges. The proposed CRF also captures the long-range color contrast, and therefore preserves the object boundaries without pre-segmenting the image; 2) we propose an efficient inference algorithm for fully connected CRFs with only one stationary constraint and show that convergence can be achieved in only a few seconds for a CRF with 5×10^9 edges. Fig. 7.1 illustrates the benefit of using our method.

7.1.1 Related Work

Context: Different contexts have been explored by many previous works in order to improve traditional CRFs. *Global image features* [73, 112] add global image information by transferring whole image labels to a test image from similar training images. This approach shows great potential when similar images can be found in the training images; otherwise, a CRF is still required to combine this information with local features. *Label consistency in a segment* is enforced by first segmenting the image and then performing labeling on the segments [31, 50, 90, 99]. To deal with the segmentation errors, the robust P^n model [50] relaxes the segment constraints by associating this information with the traditional pixel-wise CRFs. Gould et al. [36] propose a method to reshape the regions by comparing training image segments. *Top-down object detection* results are added to improve the recognition performance for structured objects [3,35,69,124,131]. *Object relationships* have been explored by previous works by modeling weak geometries over segmented regions [31, 55, 90, 97, 112]. Torralba et al. [110] captures pixel-wise relationships for the object detection task, where the goal is to obtain a rough location without precise object contour segmentations. We capture the sematic spatial interactions using the pixels and preserve the object contours at the same time; thus we have a different and harder inference problem. Moreover, we show a more principled manner to solve the pixel-level inference problem without approximations when making the updates.

Long Range CRFs: Long-range interactions in CRFs have been explored before. Sparse CRFs [70] add long-range edges at sparse locations to ensure the computational tractability. The decision tree field [87] and auto context [114]

learn the edge potentials and determine the edges to be added based on the learning results. Hierarchical CRFs [55] capture long-range interactions with a higher layer, and therefore may lose detailed information over the pixels. The co-occurrence CRF [58] encodes global object co-occurrence statistics. We propose an efficient inference algorithm for a fully connected CRF that can capture the spatial relationships among different labels.

Fast Inference with Convolution: Formulating the original problem with convolution to accelerate the computation has a long history in vision. In particular, Felzenszwalb and Huttenlocher [26,27] formulate the message passing between two nodes in belief propagation as a "min-convolution", which reduces the time from $O(K^2)$ to $O(K)$, where K is the number of categories. The assumption they rely on is that the edge costs depend only on the numerical differences of the labels. Although this assumption works well for tasks like depth estimation, it does not hold for object segmentation where the categories do not have any numerical meaning. More importantly, we focus on inference efficiency over the space domain, which is defined by the number of pixels N, while they are interested in the number of categories K. For a fully connected graph, the method still needs to pass N^2 number of messages.

7.2 Approach

We first give a brief description of the traditional 4-neighbor grid CRFs for object-based segmentation, and then describe how we augment it with spatial information and the efficient inference algorithm.

7.2.1 Grid-CRFs for Object Segmentation

The conditional random fields (CRFs) model the conditional distribution of the class labeling $\mathbf{X}$ given an image $\mathbf{I}$. We use $\mathbf{X} = \{X_1, X_2, ..., X_N\}$ to denote the set of discrete random variables with $X_i \in X$ being associated with every pixel $i \in V$, and takeing a value from the label set $L = \{l_1, ..., l_K\}$, which are the object categories. The labeling X of the image is obtained with a maximum a posteriori (MAP) estimation of the following conditional log-likelihood:

$$\log P(\mathbf{X}|\mathbf{I}) = \sum_{i \in V} \psi_i(x_i) + \sum_{i \in V, j \in N_i} \psi_{ij}(x_i, x_j) - Z(\mathbf{I}), \tag{7.1}$$

where N_i is the 4-neighbors of pixel i, and Z is a normalization term that does not depend on $\mathbf{X}$. The unary potentials $\psi_i(x_i)$ are defined with the log likelihood of variable X_i taking label x_i, which are usually computed with the texture, color, and location features extracted from a local region around i [50, 105]. The edge potential $\psi_{ij}(x_i, x_j)$ typically encodes contrast sensitive smoothness of neighboring pixels [105].

7.2.2 Fully Connected CRFs with Stationary Edges

We formulate the labeling problem with a fully connected CRF defined over the image pixels. The conditional log-likelihood is the same as the grid CRFs (Equ. 7.1), except that the edges are defined over all pairs of pixels. In other words, the neighborhood of a pixel i is defined with all other pixels $N_i = V$. The unary potential is the same as grid CRFs. The main difference is that the edge potential $\psi_{ij}(x_i, x_j)$ is a combination of the color contrast $\varphi_{ij}(x_i, x_j)$ and the spatial

relationships between two categories, $\phi_{ij}(x_i, x_j)$.

$$\psi_{ij}(x_i, x_j) = \underbrace{\phi_{ij}(x_i, x_j)}_{\text{spatial relation}} \underbrace{\varphi_{ij}(x_i, x_j)}_{\text{color contrast}} . \tag{7.2}$$

Color term: The color contrast term $\varphi_{ij}(x_i, x_j)$ encourages the same label when the colors I_i, I_j are similar, and different labels otherwise. Let $g(I_i - I_j)$ denote the gaussian function of the color difference: $g(I_i - I_j) = \exp(-\theta_c \|I_i - I_j\|^2)$. The color contrast term is defined as follows.

$$\varphi_{ij}(x_i, x_j) = \begin{cases} g(I_i - I_j) & \text{if } x_i = x_j \\ 1 - g(I_i - I_j) & \text{otherwise .} \end{cases} \tag{7.3}$$

Spatial term: The spatial term $\phi_{ij}(x_i, x_j)$ is the log-likelihood of the spatial distribution $f(x_i, x_j, p_i, p_j)$ of these two categories, i.e. the probability that two categories x_i, x_j co-occur at positions p_i, p_j. We make the assumption that this probability only depends on the relative positions of the two pixels $p_i - p_j$.

$$\phi_{ij}(x_i, x_j) = \log(\varepsilon_s + f(x_i, x_j, p_i - p_j)) \tag{7.4}$$

$$f(x_i, x_j, p_i - p_j) = \frac{1}{Z} P(x_i, x_j) P(p_i - p_j | x_i, x_j), \tag{7.5}$$

where Z is a normalization term, and $f(x_i, x_j, p_i - p_j)$ is computed as a combination of the global co-occurrence probability $P(x_i, x_j)$ and the relative position distribution given the two categories $P(p_i - p_j | x_i, x_j)$. Note that when $x_i = x_j$, the spatial term computes the likelihood that the same category occurs at a relative position. This likelihood can capture the shape and size information of the objects.

7.2.3 Quadratic Programming Relaxation

To obtain a MAP estimation of a CRF model, many inference methods have been proposed. In this paper, we adopt the quadratic programming (QP) relaxation method [98], which has been shown to perform comparable to other inference methods, such as LP relaxation [52] and tree-reweighted message passing [51].

The labeling problem that maximizes the conditional probability can be viewed as an integer program by adding a binary variable $\mu_i(x_i)$, which indicates whether a pixel i is labeled with x_i.

$$\mu_i(x_i) = \begin{cases} 1 & \text{if } X_i = x_i \\ 0 & \text{otherwise.} \end{cases} \tag{7.6}$$

With $\mu_i(x_i)$, the MAP estimation for the conditional probability $P(X|I)$ (Equ. 7.1) can be formulated as a quadratic integer program, which is further relaxed to the following quadratic program.

$$\begin{aligned} \max_{\mu} \ & \sum_{i;x_i} \psi_i(x_i)\mu_i(x_i) + \sum_{i,j;x_i,x_j} \psi_{ij}(x_i, x_j)\mu_i(x_i)\mu_j(x_j) \\ \text{s.t.} \ & \sum_{i} \mu_i(x_i) = 1 \\ & 0 \le \mu_i(x_i) \le 1 \end{aligned} \tag{7.7}$$

This QP relaxation has been proven to be a tight relaxation, and solves exactly the original MAP problem [98].

Although the optimization problem can also be formulated as a minimization problem of the Gibbs energy, which is the negative of the above objective function, the QP relaxation is usually formulated with maximization [98]. For convenience, we make the coefficients in the objective function positive by

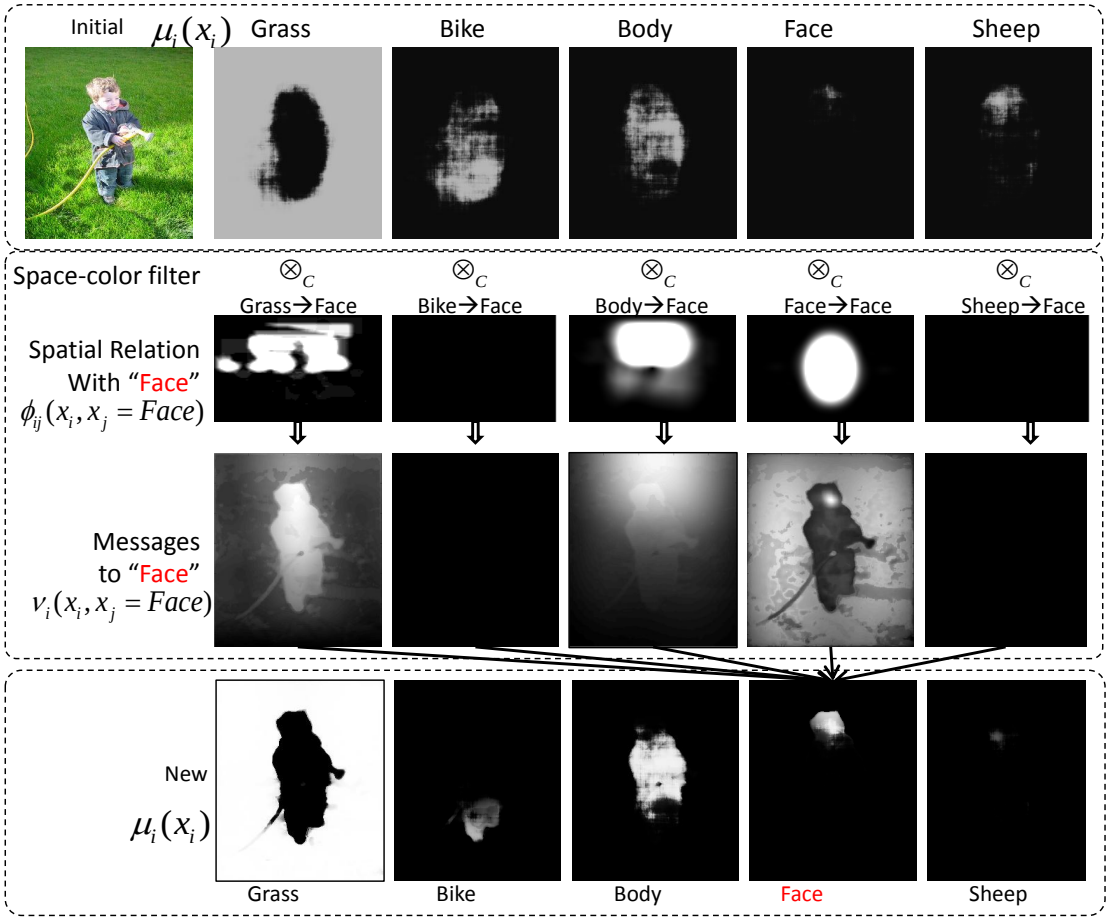


Figure 7.2: The illustration of the inference algorithm. The top row shows the initial $\mu_i(x_i)$ obtained from the unary terms. The second row shows the spatial relationships of different categories with "Face" (to save space, we show them in smaller images). The third row illustrates the messages sent to "Face" $v_i(x_i, x_j)$. The bottom row shows the updated $\mu_i(x_i)$ after normalization.

adding a constant to the log likelihood of the unary probability $\psi_i(x_i)$ and the spatial distribution $f(x_i, x_j, p_i - p_j)$ (Equ. 7.5).

7.2.4 Iterative Update Procedure

We initialize μ with the unary probabilities, and iteratively update it with the gradient of the objective function. The gradient of the objective function Q of Equ. 7.7 is as follows.

$$q_i(x_i) = \frac{\partial Q}{\partial \mu_i(x_i)} \tag{7.8}$$

$$= \psi_i(x_i) + 2 \sum_{x_j} \sum_{j} \psi_{ij}(x_i, x_j)\mu_j(x_j) \tag{7.9}$$

$$= \psi_i(x_i) + 2 \sum_{x_j} \nu_i(x_i, x_j), \tag{7.10}$$

with the symmetric edge potentials $\psi_{ij}(x_i, x_j)$.

Given the gradient $q_i(x_i)$, we can adopt the fixed point iteration to perform gradient ascent and maintain the constraints in (7.7) at the same time.

$$\mu_i^{t+1}(x_i) = \frac{\mu_i^t(x_i) q_i(x_i)}{\sum_{x_i} \mu_i^t(x_i) q_i(x_i)}, \tag{7.11}$$

where $\mu_i^t(x_i)$ is the value of $\mu_i(x_i)$ at the t^{th} iteration.

When the edge potentials do not define a negative definite matrix, gradient ascent may converge to a local maximum, as other iterative update methods, such as max-product belief propagation or mean field. While we can follow [98] to change the values on the diagonal of the matrix to make a convex approximation, we found the original edge potentials produce a reasonable result in practice.

7.2.5 Gradient as Image Filtering

The main bottleneck for computing the gradient (Equ. 7.10) of a fully connected CRF is the computation for $\nu_i(x_i, x_j)$, which is the weighted summation of the messages from all other pixels to i when the categories are x_i, x_j. A naive implementation would have computational complexity $O(N^2)$ for computing this term for all pixels, where N is the number of pixels. Combining with the edge potential definitions (Equ. 7.2, 7.3, and 7.5), for two categories (x_i, x_j), $\nu_i(x_i, x_j)$ is calculated as follows. For simplicity, we omit x_i, x_j in the variables.

$$\begin{aligned} \nu_i &= \sum_j \underbrace{\phi_{ij}(x_i, x_j)}_{\text{spatial}} \underbrace{\varphi_{ij}(x_i, x_j)}_{\text{color}} \mu_j \\ &= \begin{cases} \sum_j \phi(p_i - p_j) g(I_i - I_j) \mu_j & \text{if } x_i = x_j \\ \sum_j \phi(p_i - p_j) \Big(1 - g(I_i - I_j)\Big) \mu_j & \text{otherwise .} \end{cases} \end{aligned} \tag{7.12}$$

The intuition of this formulation is that for the same category, we prefer propagating the messages to similar color pixels, while for different categories, we propagate the messages to pixels of different colors, which are more likely to come from different objects (illustrated Fig. 7.2).

Since $\phi(p_i - p_j)$ only depends on the relative position $p_i - p_j$, if we do not have the color term, the above equation can be solved for all pixels i by treating $\phi(p_i - p_j)$ as a filter for an image valued with $\mu_i(x_i)$. Image filtering can be greatly accelerated with Fast Fourier Transform (FFT), which reduces the complexity to $O(N \log N)$.

However, if we have the color term, the equation does not take the form of convolution. Instead, it is a filtering on the space and color dimension at the same, where the color filter is a Gaussian. Image filtering on both space

and color dimensions has been studied before, as the edge preserving image smoothing problem, which is also called Bilateral Filtering [23,91,108]. In bilateral filtering, the spatial filter is also a Gaussian. Although we have a different space term, we can borrow the idea from the algorithms proposed for this problem.

Borrowing the idea from [23], if we fix the color value I_i for the pixel i, Equ. 7.12 will become a convolution of the function $H_j^c = g(I_c - I_j)\mu_j$. Therefore, we discretize the color values into C clusters $\{I^c\}$, and compute a linear filtering for each I_c.

$$\nu_i^c = \sum_j \phi(p_i - p_j) g(I_c - I_j)\mu_j \tag{7.13}$$

$$= \sum_j \phi(p_i - p_j) H_j^c, \tag{7.14}$$

when $x_i = x_j$. The final output ν_i is a linear interpolation between the output ν_i^c of two closest values I_c of I_i.

The resulting computation complexity is $O(CN \log N)$, where C is the number of clusters we create for an image. In practice, we found $10 - 15$ clusters produce good results on most images. We also adopt the down-sampling scheme as described in [23] to further speed up the computation (detailed in the next section). On a 213×320 resolution image, computing the gradient (Equ. 7.10) for all pixels takes less than 0.1 seconds. Although further acceleration can be expected with more recent algorithms on the bilateral filtering problem [91], we leave deep exploration on this line for future work.

The illustration of the iterative update is shown in Fig. 7.2

7.2.6 Learning

We perform piecewise training as previous works [50,58,105], which learn each potentials with ground truth instead of a joint learning of all potentials at the same time. For the unary term, we employ the same parameters as [50].

For the edge potential, we need to learn the co-occurrence distribution $P(l, l')$ of two categories l, l' and their relative spatial distribution $P(dp_x, dp_y|l, l')$; we use (dp_x, dp_y) to denote the position offset on x, y axes. We perform a maximum likelihood estimation for these distributions using the training images. The co-occurrence distribution can be easily obtained by counting the number of times categories l, l' appear in the same image. To compute $P(dp_x, dp_y|l, l')$, for each image which has objects l, l', we count the frequency l and l' occur at relative positions (dp_x, dp_y). This can be efficiently computed with a cross-correlation over the pixel-wise ground truth for categories l and l' (Fig. 7.3). Note that the order of l and l' matters. Learning the distributions on 276 images for each pair of 21 categories takes less than 25 seconds.

To avoid over-fitting, we compute a quantized spatial distribution. Specifically, rather than computing pixel-wise $P(dp_x, dp_y|l, l')$, we compute a binned relative spatial distribution with $M_x \times M_y$ bins. In practice, we use step size 5 for both x and y axes. The quantization also enables us to perform image filtering (previous subsection) using the down-sampling scheme as [23] with little quality loss. Specifically, we do filtering on the down-sampled image space, but perform the final interpolation with full-scale images. Note that this is quite different from performing inference on a resized input image, which will lose detailed information. We refer the readers to [23] for the detailed algorithm. With the down-sampling scheme, the computational complexity for computing

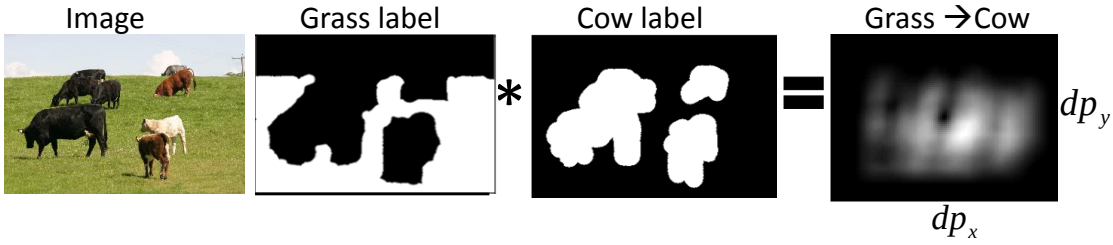


Figure 7.3: The illustration for computing the frequency that the pixels labeled with "grass" and "cow" appear at different relative positions (dp_x, dp_y). The rightmost image shows the resulting (dp_x, dp_y) space, which can be obtained with a cross-correlation from the ground truth of the two categories.

the gradients reduces to $O(CM_xM_yl\log(M_xM_y))$, where C is the number of color clusters created for an image.

Other model parameters, such as different weights given to unary and edge potentials, are manually set to minimize the segmentation errors on a validation set.

7.2.7 Practical Issues

At each iteration, we need to do the image filtering (section 7.2.5) for every pair of categories. We can first compute the Fourier transform for the scores of all categories to avoid K^2 times of FFT computation, where K is the number of categories. Moreover, when the co-occurrence probability of two categories is very low, we do not make propagation between them. Finally, we pre-compute the Fourier transforms for the edge potentials to avoid recomputing them at every iteration.

7.3 Experiments

We perform the experiments on the Sowerby-7 [38] and MSER-23 [105] datasets. The Sowerby dataset contains 7 object classes of 104 images, and the MSRC dataset has 23 object classes of 591 images. We compare favorably with the state-of-the-art object segmentation approaches, including 4-neighbor grid CRFs [105], robust P^n CRFs [50], and the robust P^n plus object co-occurrence CRFs [58]. For implementation of these methods, we use the publicly available code provided by the authors. We consider our baseline as different inference methods where the same low level features are used, and treat other works that encode additional features, such as features from the whole image [73, 112] or image segments [57], as complementary to ours.

7.3.1 Synthetic Data

We first perform an experiment on the synthetic data to evaluate the proposed inference algorithm. In this experiment, we would like to know how well the proposed fully connected CRF work when we do not have reliable unary potentials but know the exact spatial object relationships. For a selected 213x320 resolution image from the MSRC dataset, we learn relative spatial distribution between different objects using the ground truth of the same image. For the unary potentials, we randomly generate the class probabilities for each pixel from a uniform distribution over all categories included in the image.

Analysis: The inference results of the synthetic data are shown in Fig. 7.4. No surprisingly, using the 4-neighbor grid CRF, where no unary and spatial

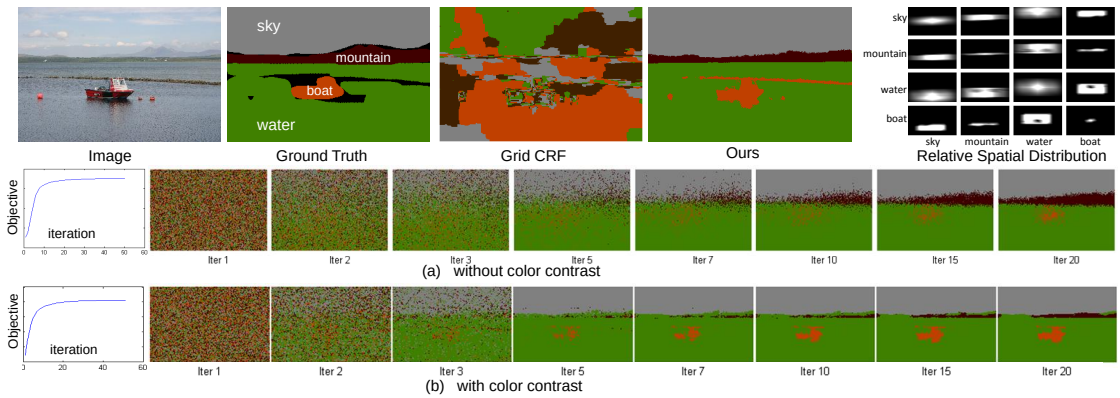


Figure 7.4: Analysis on the synthetic data where no unary cues are used. The rightmost image on the top row shows the relative spatial distribution of each pair of categories. The middle and bottom rows show the performance of our method with and without color contrast term in the edge potentials respectively. The leftmost images of these two rows plot the changes of the objective values in the QP relaxation at each iteration. The rest images show the changes of the predicted labels as we perform iterative updates.

cues are available, we cannot locate the objects correctly. With relative spatial distribution, we can predict a reasonable layout of the objects, but do not preserve good object boundaries when color contrasts are not included. With color contrast in the edge potentials, we can provide reliable object segments. The QP optimization converges after around 20-30 iterations in both cases. In all the following experiments, we always set the maximum iteration to 30.

Running time: We compare the running of our method with max-product loopy belief propagation (BP) in Fig. 7.5. We implemented the proposed method using Matlab with some C++ help on a server with Quad-Core 2.66G Intel Xeon CPU and 12G memory. We vary the number of neighborhood pixels each node connects to in the CRF. The running time of BP for each iteration is linear to neighborhood size, while the proposed inference algorithm is almost constant.

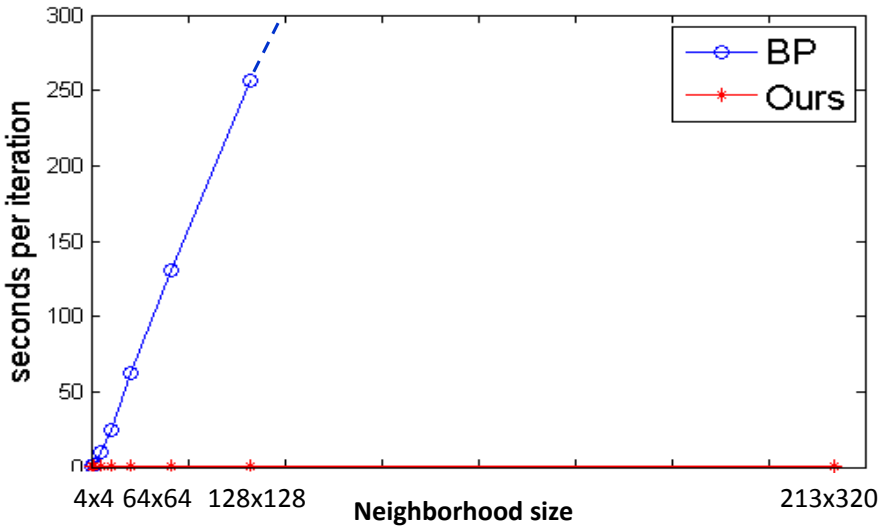


Figure 7.5: Average running time (seconds) per iteration for max-product belief propagation and the proposed method for a CRF defined over 213x320 pixels with edges over different neighborhood size.

The reason is that the main computation of the proposed algorithm is a FFT over the image, which is $O(N \log N)$ regardless of the neighborhood size (N is the total number of pixels in the image). When all pixel pairs are connected, BP runs out of memory on our computer, while the estimated running time would be more than one hour even if enough memory is provided. In comparison, our algorithm is less than 0.1 second.

7.3.2 Segmentation without Unary Cues

Now we ask ourselves the question: in real cases, if we know what objects are present in an image, how well can we perform object localization and segmentation with only the relative positions of different categories. That is, we do not use any classifiers of the low level cues, such as texture or color, which can be computationally expensive for both training and testing. Instead, we only learn

the relative spatial distributions between two categories. We experiment on the Sowerby dataset [38], where we can safely assume all images have the same categories: *sky*, *road*, *vegetable*, and *building*. We ignore the *car* and *sign* categories which are only present in a few images. We randomly select 50% images to train the spatial distribution, and use the rest 50% for testing.

Qualitative analysis: Fig. 7.6 illustrates the predicted segmentation results of example testing images using different methods. We first compare our method with 4-neighbor grid CRF when randomly generated unary potentials are used. Since the grid CRF only has neighboring color contrast information, the predicted labeling is quite random. To make fair comparison, we learn absolute location distribution for different classes and encode it as the unary potential. With the location information, the grid CRF performs better, but provides similar prediction for all images. On the contrary, the proposed fully connected CRF can make reasonable object arrangements and preserve object boundaries for different test images. We can explain our method as follows: the spatial relationship term in the edge potentials predicts a rough object layout, and the color contrast term among all pixel pairs enforces a better object segmentation. The bottom row of Fig. 7.6 shows a typical error made by our algorithm, when the input image has an usual spatial layout among objects.

Quantitative comparison: Table 7.1 presents the pixel-wise accuracy using different methods. In addition to the grid CRF, we also compare with the robust P^n CRF [50], which encourages the label consistency within each segmented region.

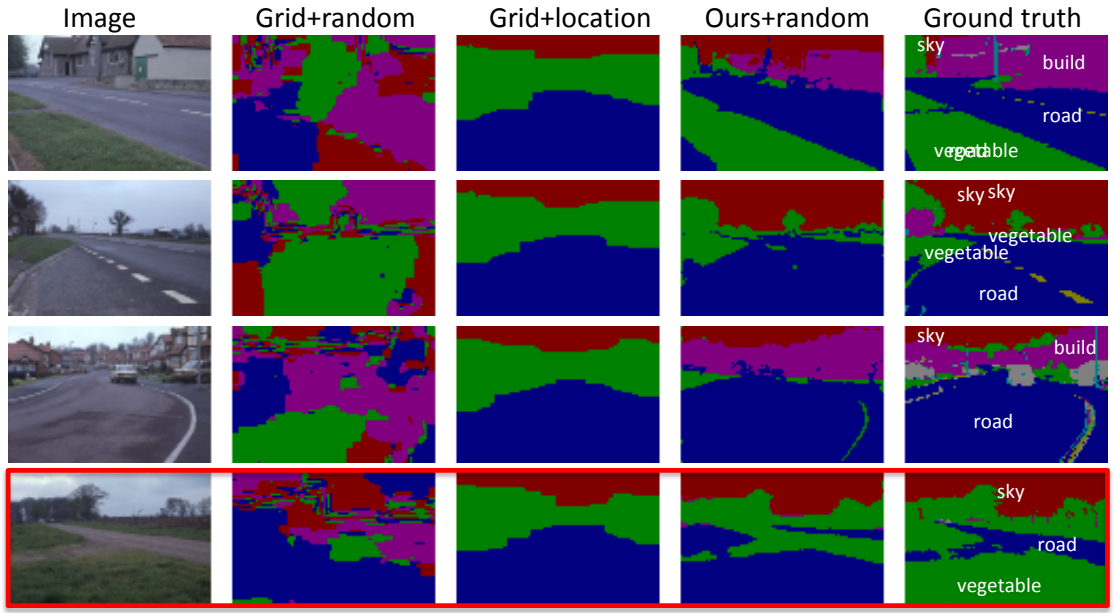


Figure 7.6: Qualitative results on the Sowerby dataset when only absolute or relative locations are learned during training. From left to right, we show the test image, grid CRF with random unary potential, grid CRF with learned location potentials, proposed method with random unary potential, and the ground truth.

	Grid+random	Grid+loc	P^n+loc	Ours+random	Ours+loc
Global	22.2	68.2	62.1	76.4	**78.9**
Avg. Recall	24.0	54.5	63.0	67.8	**70.8**
Avg. Precision	21.8	49.4	61.7	67.7	**70.7**

Table 7.1: Quantitative results on the Sowerby dataset when only absolute or relative locations are learned during training. We show the pixel-wise global accuracy (%) over all images, and recall (%) and precision (%) averaged over different categories.

7.3.3 Segmentation with Unary Cues

Finally, we show the experiment results on the MSRC-23 dataset when more unary information is used. We use the same experiment settings as [105]: 45% of the images for training, 10% for validation, and the rest 45% for testing. Following [105], we ignore the *mountain* and *horse* categories which have too few training images. For the unary potentials, we use the same texture, color, and

location classifiers as [50] by applying their published code.

Qualitative analysis: Fig. 7.7 illustrates the labeling results of example test images using different CRFs. Grid CRF does not capture class relationships, and prefers smoother object boundaries. Incorporating object co-occurrence statistics [58] in the Robust P^n CRF [50] provides better object boundaries and remove the objects that rarely co-occur in the same image, eg. *cat* and *book* (first row). The proposed CRF incorporates both object co-occurrence information and their spatial arrangements, and produce better class predictions. For example, *road* is more likely to appear below *grass* than *water* (first row); *boat* has higher possibility to appear right above *water* than *building* (third row); and *car* rarely appears right above *grass* (bottom row). Our approach benefits from modeling everything in a unified CRF, and thus is able to make the object layout adjustment only when the unary potentials are more confused. The second row in the figure is an example where the color contrast and spatial relation together leads to a better prediction. Since *sky* is very likely to appear above *grass*, and the colors are quite different for a particular region, we can make the correct changes. Moreover, with the long-range color contrast, we can also provide more precise object boundaries, eg. *bird* in the first row, and *tree* in the bottom row.

Quantitative comparison: Table 7.2 compares different CRFs with the recognition rates of different categories. Our method improves previous works on most of the categories by 1% to 14%. Note that the ground truth provided by the MSRC dataset is quite rough, and therefore more precise object boundaries may not reflect better accuracies. Table 7.3 presents the global accuracies over all images, and precision and recall averaged over different categories. Robust P^n+co-occurrence [58] performs better than Robust P^n by incorporating global

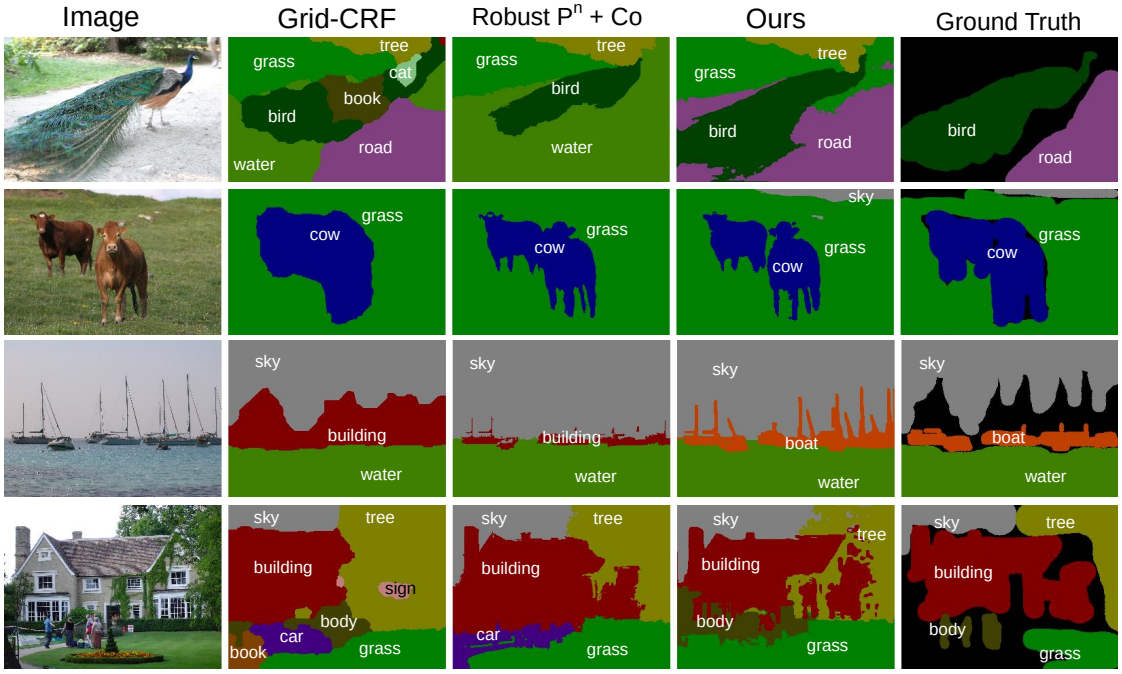


Figure 7.7: Results on the MSRC dataset with different CRFs

co-occurrence statistics. By further encoding spatial relationships, we improve the Robust P^n+co-occurrence by 1.9% in global accuracy. Since the global accuracy is biased for categories that have more pixels, such as *grass* or *tree*, more improvement is observed for average precision (3.6%) and recall (2.7%). The proposed method also runs faster than Robust P^n+co-occurrence. Excluding the time for obtaining the unary potentials, the proposed method process an image in 2 to 8 seconds, while robust P^n+co-occurrence [58] requires 8 to 30 seconds per image. Gaussian CRF [54] also captures long-range color contrast. By incorporating spatial interactions among different object categories, we improve the method by 1.0% in global accuracy and 2.4% in average recall.

Errors: Fig. 7.8 shows example errors made by our method. When we do not have a good support from the unary potential, we predict the labels with

	building	grass	tree	cow	sheep	sky	plane	water	face	car
Unary	68.8	96.5	**90.1**	82.9	85.0	94.2	**87.9**	80.4	88.7	77.0
Grid-CRF	69.0	96.8	88.6	82.9	85.9	94.6	87.4	81.8	88.2	78.0
Robust P^n [50]	70.0	96.8	89.7	83.9	86.9	94.6	87.4	81.8	89.1	78.0
Robust P^n + Co [58]	71.5	97.0	89.8	84.7	**87.7**	94.8	87.4	81.9	88.9	78.9
Ours	**75.4**	**98.1**	88.7	**86.1**	86.2	**96.3**	87.5	**85.0**	**93.9**	**79.1**

	bike	flower	sign	bird	book	chair	road	cat	dog	body	boat
Unary	92.5	86.1	62.1	41.5	93.8	66.5	86.3	81.0	53.9	75.0	30.6
Grid-CRF	92.9	87.0	63.5	41.6	94.1	66.3	86.0	81.3	54.2	75.0	30.2
Robust P^n [50]	92.7	88.0	62.5	41.2	94.1	66.3	87.0	82.3	**54.6**	75.7	29.2
Robust P^n + Co [58]	93.6	89.0	62.6	42.1	94.6	66.8	87.5	82.5	51.9	76.5	28.8
Ours	**93.7**	**90.0**	**64.2**	**55.9**	**94.8**	**81.6**	**88.2**	**85.4**	53.6	**79.1**	**32.1**

Table 7.2: Recognition rates (%) of different categories in the MSRC dataset. Recognition rate is computed as the recall of each category.

	Unary	Grid	P^n [50]	P^n + Co [58]	Gaussian [54]	Ours
Global	84.1	84.6	84.7	85.1	86.0	**87.0**
Avg. Precision	79.3	80.5	80.6	81.9	n/a	**85.5**
Avg. Recall	77.1	77.4	77.7	78.0	78.3	**80.7**

Table 7.3: Performance comparison on the MSRC dataset with pixel-wise global accuracies (%) over all images, precision (%) and recall (%) averaged over different categories. Except the Gaussian CRF [54], we show the performance using the code by [58]. Therefore, exactly the same unary potentials are used. For [54], we cite the performance in their paper.

more common spatial arrangements, which can be wrong in some cases.

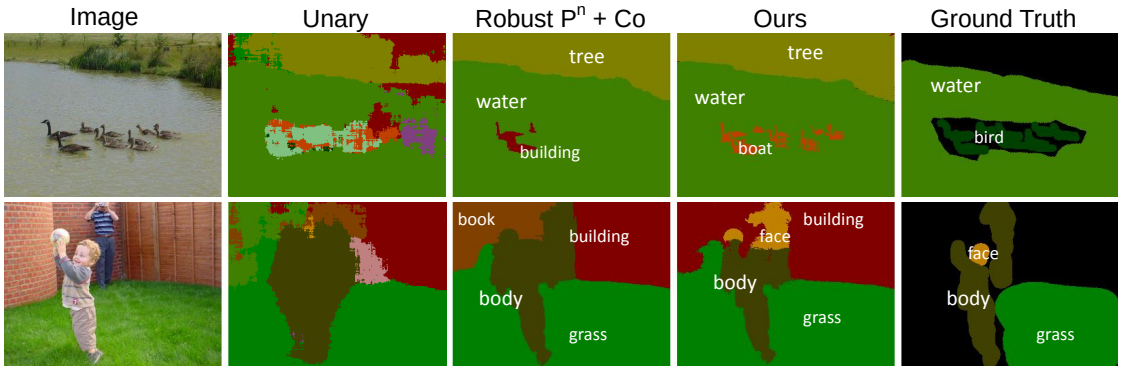


Figure 7.8: Example errors made by our approach on the MSRC dataset.

CHAPTER 8

CONCLUSION

For most vision techniques, there is a trade-off between the amount of the spatial information modeled and the computational cost of the technique. In this thesis, we targeted algorithms that model more spatial information while limiting the sacrifice of the computational cost. We have focused on two vision problems : the image/video representation and the pixel-level image labeling.

8.1 Image and Video Representation

We proposed an algorithm that encodes geometry information to the Bag-of-Words (BoW) model with high-order features (HOF). Our algorithm can capture both short and long range spatial information in the HOFs and compute one to *infinite* order features in time linear to the number of features per image (same computational complexity as BoW). The algorithm is general enough to be applied to different vision applications where BoW was used.

For the application of object recognition, we used the proposed algorithm to compute a SVM kernel to perform classification with SVM on the HOF space. To further localize the objects, we proposed an algorithm that avoids computing the HOF kernel for every sub-window of an image. We have also integrated the HOF based method with the structure learning framework, which directly optimizes the localization performance when training the detectors. The experiment results show that using HOFs to model the object shapes outperforms the accuracies of BoW.

For image retrieval, where efficiency is even more crucial, we integrated the proposed HOF algorithm to the inverted index structure, and thus can model spatial information at the searching step of a retrieval system. The experiment results showed that our approach outperforms the state-of-arts, which use the BoW model plus a RANSAC post-processing, while requiring similar memory usage and much less computational time than the system based on BoW+RANSAC.

Finally, we showed how to use the HOFs in videos. The proposed algorithm was extended by adding a temporal dimension to capture the causality relationships of the local movements. Experiment results demonstrated that our approach improves the state-of-the-art approaches in activity recognition, and is widely applicable to a large variety of activities, as illustrated by the diversity of experimental datasets.

8.1.1 Future Work

Other applications: We have already presented four different applications using our HOF algorithm. We would like to use our algorithm to other vision applications which also rely on the local features. We are also interested in adaptations of the algorithm for different types of images, such as the ariel/satellite photos or bioimages. For example, rotation invariance may be important when detecting objects from ariel photos; therefore, we should also add another *angle* dimension to the correspondence-transform algorithm (Chapter 2).

Other usage of the algorithm: The algorithm for finding the co-occurring HOFs in two images is essentially a method for detecting spatially consistent

components between images efficiently. It can also be used in places other than the HOF framework. Li et al. [68] have already used the method for identifying spatially consistent object-sets in order to create the visual phrases, such as "people riding a horse", "people sitting on the sofa", and "5-piece dining table set". We would like to explore more usages of the algorithm.

8.2 Pixel-Level Image Labeling

We introduced a fully connected CRF which encodes spatial relationships among different objects and preserves object contours at the same time. We proposed an efficient algorithm to inference the fully connected CRF. The experiment results demonstrated the efficiency and effectiveness of the proposed CRF.

8.2.1 Future Work

Learning the CRF: We have focused on the inference algorithm of the fully connected CRFs, while for the training part, we performed piecewise training as many other previous works [50, 58, 105]. We can further improve the training of the CRF model with the structured learning method [113] by formulating the output of the structured SVM as the pixel labeling of the input image. The structured learning can learn the parameters of the CRF by directly optimizing labeling accuracies. The key requirement for applying the structured learning method is an efficient inference algorithm, because it performs the inference multiple times on each image during training. Since the proposed inference al-

gorithm is very efficient, we can easily apply the structured learning method to train the fully-connected CRF. Thus we can learn all parameters of the fully-connected CRF as a whole, and learn the relative spatial relationships between objects discriminatively with a maximum-margin objective function. Thus, we can expect more improvements in the performance of the labeling.

Other inference algorithm: The efficiency of the proposed inference algorithm is mainly resulted by formulating the gradient of the quadratic programming (QP) relaxation as a fast fourier transform (FFT) procedure. We have used the iterative update procedure to optimize the QP relaxation. Other more sophisticated gradient descent method for the QP relaxation [98] can be considered too, as long as the gradient can still be formulated as FFT. Moreover, we are also interested to see whether other inference algorithms, such as mean-field or tree decomposition, can also be accelerated similarly with FFT.

Other applications: We have used the fully-connected CRF for the object-based segmentation task. The proposed algorithm is also applicable to any other application as long as the stationary assumption holds. We are interested in investigating other applications where incorporating relative spatial information among the labels are beneficial.

APPENDIX A

RELATED PUBLICATIONS

The following are the publications related to this thesis.

1. Yimeng Zhang, Xiaoming Liu, Ming-Ching Chang, Weina Ge, and Tsuhan Chen, "Spatial-Temporal Phrases for Activity Recognition", in *ECCV*, 2012
2. Yimeng Zhang and Tsuhan Chen, "Efficient Inference for Fully-Connected CRFs", in *CVPR*, 2012.
3. Yimeng Zhang, Weina Ge, Ming-Ching Chang, and Xiaoming Liu, "Group Context Learning for Event Recognition", in *WACV*, 2012
4. Yimeng Zhang, Zhaoyin Jia, and Tsuhan Chen. "Image Retrieval with Geometry Preserving Visual Phrase", in *CVPR*, 2011
5. Yimeng Zhang and Tsuhan Chen. "Implicit Shape Kernel for Discriminative Learning of the Hough Transform Detector.", in *BMVC*, 2010.
6. Yimeng Zhang and Tsuhan Chen. "Weakly Supervised Object Recognition and Localization with Invariant High Order Features.", in *BMVC*, 2010
7. Yimeng Zhang and Tsuhan Chen. "Efficient Kernels for Identifying Unbounded Order Spatial Features.", in *CVPR*, 2009.

BIBLIOGRAPHY

[1] S. Ali, A. Basharat, and M. Shah. Chaotic invariants for human action recognition. In *ICCV*, 2007.

[2] Francis R. Bach, Gert R. G. Lanckriet, and Michael I. Jordan. Multiple kernel learning, conic duality, and the SMO algorithm. In *ICML*, 2004.

[3] Luca Bertelli, Tianli Yu, Diem Vu, and Burak Gokturk. Kernelized structural SVM learning for supervised object segmentation. In *CVPR*, 2011.

[4] M. B. Blaschko and C. H. Lampert. Learning to localize objects with structured output regression. In *ECCV*, 2008.

[5] M. B. Blaschko and C. H. Lampert. Object localization with global and local context kernels. In *BMVC*, 2009.

[6] O. Boiman, E. Shechtman, and M. Irani. In defense of nearest-neighbor based image classification. In *CVPR*, 2008.

[7] William Brendel and Sinisa Todorovic. Learning spatiotemporal graphs of human activities. In *ICCV*, 2011.

[8] Yang Cao, Changhu Wang, Zhiwei Li, Liqing Zhang, and Lei Zhang. Spatial-bag-of-features. In *CVPR*, 2010.

[9] D. Chaitanya, R. Deva, and F. Charless. Discriminative models for multi-class object layout. In *ICCV*, 2009.

[10] Ming-Ching Chang, Nils Krahnstoever, and Weina Ge. Probabilistic group-level motion analysis and scenario recognition. In *ICCV*, 2011.

[11] O. Chum and A. Zisserman. An exemplar model for learning object classes. In *CVPR*, 2007.

[12] Ondrej Chum and Jiri Matas. Unsupervised discovery of co-occurrence in sparse high dimensional data. In *CVPR*, 2010.

[13] Ondrej Chum, Michal Perdoch, and Jiri Matas. Geometric min-hashing: finding a (trick) needle in a haystack. In *CVPR*, 2009.

[14] Ondrej Chum, James Philbin, Michael Isard, and Andrew Zisserman. Scalable near identical image and shot detection. In *CIVR*, 2007.

[15] Ondrej Chum, James Philbin, Josef Sivic, Michael Isard, and Andrew Zisserman. Total recall: automatic query expansion with a generative feature model for object retrieval. In *ICCV*, 2007.

[16] Ondrej Chum, James Philbin, and Andrew Zisserman. Near duplicate image detection: min-hash and tf-idf weighting. In *BMVC*, 2008.

[17] G. Csurka, C. R. Dance, L. Fan, J. Willamowski, and C. Bray. Visual categorization with bags of keypoints. In *ECCV*, 2004.

[18] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, 2005.

[19] Chaitanya Desai, Deva Ramanan, and Charless C. Fowlkes. Discriminative models for multi-class object layout. *IJCV*, 95(1):1–12, 2011.

[20] P. Dollar, V. Rabaud, G. Cottrell, and S. J. Belongie. Behavior recognition via sparse spatio-temporal features. In *PETS Workshop*, 2005.

[21] Lixin Duan, Dong Xu, Ivor Wai-Hung Tsang, and Jiebo Luo. Visual event recognition in videos by learning from web data. In *CVPR*, 2010.

[22] Olivier Duchenne, Armand Joulin, and Jean Ponce. A graph-matching kernel for object categorization. In *ICCV*, 2011.

[23] Frédo Durand and Julie Dorsey. Fast bilateral filtering for the display of high-dynamic-range images. In *SIGGRAPH*, 2002.

[24] A. A. Efros, A. C. Berg, G. Mori, and J. Malik. Recognizing action at a distance. In *ICCV*, 2003.

[25] P. Felzenszwalb, D. McAllester, and D. Ramanan. A discriminatively trained, multiscale, deformable part model. In *CVPR*, 2008.

[26] Pedro F. Felzenszwalb and Daniel P. Huttenlocher. Pictorial structures for object recognition. *IJCV*, 61(1):55–79, 2005.

[27] Pedro F. Felzenszwalb and Daniel P. Huttenlocher. Efficient belief propagation for early vision. *IJCV*, 70(1):41–54, 2006.

[28] Jiashi Feng, Bingbing Ni, Qi Tian, and Shuicheng Yan. Geometric ℓp-norm feature pooling for image classification. In *CVPR*, 2011.

[29] R. Fergus, P. Perona, and A. Zisserman. Weakly supervised scale-invariant learning of models for visual recognition. *International Journal of Computer Vision*, 71(3):273–303, 2007.

[30] V. Ferrari, L. F., F. Jurie, and C. Schmid. Groups of adjacent contour segments for object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(1):36–51, 2008.

[31] Carolina Galleguillos, Andrew Rabinovich, and Serge Belongie. Object categorization using co-occurrence, location and appearance. In *CVPR*, 2008.

[32] U. Gaur, Y. Zhu, B. Song, and A. Roy-Chowdhury. A "string of feature graphs" model for recognition of complex activities in natural videos. In *ICCV*, 2011.

[33] Peter V. Gehler and Sebastian Nowozin. On feature combination for multiclass object classification. In *ICCV*, 2009.

[34] A. Gilbert, J. Illingworth, and R. Bowden. Scale invariant action recognition using compound features mined from dense spatio-temporal corners. In *ECCV*, 2008.

[35] Josep M. Gonfaus, Xavier Boix Bosch, Joost van de Weijer, Andrew D. Bagdanov, Joan Serrat Gual, and Jordi Gonzàlez Sabaté. Harmony potentials for joint classification and segmentation. In *CVPR*, 2010.

[36] Stephen Gould, Richard Fulton, and Daphne Koller. Decomposing a scene into geometric and semantically consistent regions. In *ICCV*, 2009.

[37] Derek Greene and Padraig Cunningham. Practical solutions to the problem of diagonal dominance in kernel document clustering. In *ICML*, volume 148, 2006.

[38] X. M. He, R. S. Zemel, and D. Ray. Learning and incorporating top-down cues in image segmentation. In *ECCV*, pages I: 338–351, 2006.

[39] G. Heitz and D. Koller. Learning spatial context: Using stuff to find things. In *ECCV*, 2008.

[40] Timothy M. Hospedales, Shaogang Gong, and Tao Xiang. A Markov clustering topic model for mining behaviour in video. In *ICCV*, 2009.

[41] Adrian Ion, Joao Carreira, and Cristian Sminchisescu. Probabilistic joint image segmentation and labeling. In *NIPS*, 2011.

[42] Herve Jegou, Matthijs Douze, and Cordelia Schmid. Hamming embedding and weak geometric consistency for lage scale image search. In *ECCV*, 2008.

[43] Herve Jegou, Matthijs Douze, and Cordelia Schmid. Packing bag-of-features. In *ICCV*, 2009.

[44] Herve Jegou, Matthijs Douze, Cordelia Schmid, and Patrick Perez. Aggregating local descriptors into a compact image representation. In *CVPR*, 2010.

[45] Yangqing Jia, Chang Huang, and Trevor Darrell. Beyond spatial pyramids: receptive field learning for pooled image features. In *CVPR*, 2012.

[46] T. Joachims. *Making large-scale support vector machine learning practical*. MIT Press, 1999.

[47] Sergey Karayev, Mario Fritz, Sanja Fidler, and Trevor Darrell. A probabilistic model for recursive factorized image features. In *CVPR*, 2011.

[48] Y. Ke, R. Sukthankar, and M. Hebert. Event detection in crowded videos. In *ICCV*, 2007.

[49] J. Kim and K. Grauman. Observe locally, infer globally: A space-time MRF for detecting abnormal activities with incremental updates. In *CVPR*, 2009.

[50] Pushmeet Kohli, Lubor Ladicky, and Philip H. S. Torr. Robust higher order potentials for enforcing label consistency. In *CVPR*, 2008.

[51] Vladimir Kolmogorov. Convergent tree-reweighted message passing for energy minimization. *PAMI*, 28(10):1568–1583, 2006.

[52] Nikos Komodakis and Nikos Paragios. Beyond loose lp-relaxations: Optimizing mrfs by repairing cycles. In *ECCV*, 2008.

[53] Adriana Kovashka and Kristen Grauman. Learning a hierarchy of discriminative space-time neighborhood features for human action recognition. In *CVPR*, 2010.

[54] Philipp Krahenbuhl and Vladlen Koltun. Efficent inference in fully connected crfs with gaussian edge potentials. In *NIPS*, 2011.

[55] Sanjiv Kumar and Martial Hebert. A hierarchical field framework for unified context-based classification. In *ICCV*, 2005.

[56] Daniel Küttel, Michael D. Breitenstein, Luc J. Van Gool, and Vittorio Ferrari. What's going on? discovering spatio-temporal dependencies in dynamic scenes. In *CVPR*, 2010.

[57] Lubor Ladicky, Christopher Russell, Pushmeet Kohli, and Philip H. S. Torr. Associative hierarchical crfs for object class image segmentation. In *ICCV*, 2009.

[58] Lubor Ladicky, Christopher Russell, Pushmeet Kohli, and Philip H. S. Torr. Graph cut based inference with co-occurrence statistics. In *ECCV*, 2010.

[59] C. H. Lampert, M. B. Blaschko, and T. Hofmann. Beyond sliding windows: Object localization by efficient subwindow search. In *CVPR*, 2008.

[60] Christoph H. Lampert. Detecting objecs in large image collections and videos by efficient subimage retrieval. In *ICCV*, 2009.

[61] I. Laptev and T. Lindeberg. Space-time interest points. In *ICCV*, 2003.

[62] I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld. Learning realistic human actions from movies. In *CVPR*, 2008.

[63] S. Lazebnik and M. Raginsky. An empirical bayes approach to contextual region classification. In *CVPR*, 2009.

[64] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *CVPR*, 2006.

[65] Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. Beyond bags of fea-

tures: spatial pyramid matching for recognizing natural scene categories. In *CVPR*, 2006.

[66] B. Leibe, A. Leonardis, and B. Schiele. Robust object detection with interleaved categorization and segmentation. *International Journal of Computer Vision*, 77(1-3):259–289, 2008.

[67] M. Leordeanu, M. Hebert, and R. Sukthankar. Beyond local appearance: Category recognition from pairwise interactions of simple features. In *CVPR*, June 2007.

[68] Congcong Li, Devi Parikh, and Tsuhan Chen. Automatic discovery of groups of objects for scene understanding. In *CVPR*, 2012.

[69] Fuxin Li, João Carreira, and Cristian Sminchisescu. Object recognition as ranking holistic figure-ground hypotheses. In *CVPR*, 2010.

[70] Yunpeng Li and Daniel P. Huttenlocher. Sparse long-range random field and its application to image denoising. In *ECCV*, 2008.

[71] Zhe Lin and Jonathan Brandt. A local bag-of-features model for large scale object retrieval. In *ECCV*, 2010.

[72] H. Ling and S. Soatto. Proximity distribution kernels for geometric context in category recognition. In *ICCV*, 2007.

[73] Ce Liu, Jenny Yuen, and Antonio Torralba. Nonparametric scene parsing: Label transfer via dense scene alignment. In *CVPR*, 2009.

[74] D. Liu, G. Hua, P. Viola, and T. Chen. Integrated feature selection and higher-order spatial feature extraction for object categorization. In *CVPR*, 2008.

[75] J. G. Liu, J. B. Luo, and M. Shah. Recognizing realistic actions from videos "in the wild". In *CVPR*, 2009.

[76] J. G. Liu and M. Shah. Learning human actions via information maximization. In *CVPR*, 2008.

[77] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60:91–110, 2004.

[78] David G. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60:91–110, November 2004.

[79] L. Zelnik Manor and M. Irani. Statistical analysis of dynamic actions. *PAMI*, 28(9):1530–1535, 2006.

[80] Ross Messing, Chris Pal, and Henry A. Kautz. Activity recognition using the velocity histories of tracked keypoints. In *ICCV*, 2009.

[81] K. Mikolajczyk, B. Leibe, and B. Schiele. Multiple object class detection with a generative model. In *CVPR*, 2006.

[82] K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(10):1615–1630, 2005.

[83] Krystian Mikolajczyk and Cordelia Schmid. Scale & affine invariant interest point detectors. 2004.

[84] Fabian Nater, Helmut Grabner, and Luc J. Van Gool. Exploiting simple hierarchies for unsupervised human behavior analysis. In *CVPR*, 2010.

[85] David Nister and Henrik Stewenius. Scalable recognition with a vocabulary tree. In *CVPR*, 2006.

[86] S. Nowozin, G. Bakir, and K. Tsuda. Discriminative subsequence mining for action classification. In *ICCV*, 2007.

[87] Sebastian Nowozin, Carsten Rother, Shai Bagon, Toby Sharp, Bangpeng Yao, and Pushmeet Kohli. Decision tree fields. In *ICCV*, 2011.

[88] A. Opelt, A. Pinz, M. Fussenegger, and P. Auer. Generic object recognition with boosting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2006.

[89] S. E. Palmer. The effect of contextual scenes on the identification of objects. *Memory and Cognition*, 3:519–526, 1975.

[90] Devi Parikh, C. Lawrence Zitnick, and Tsuhan Chen. From appearance to context-based recognition: Dense labeling in small images. In *CVPR*, 2008.

[91] Sylvain Paris and Frédo Durand. A fast approximation of the bilateral filter using a signal processing approach. *IJCV*, 81(1):24–52, 2009.

[92] Michal Perdoch, Ondrej Chum, and Jiri Matas. Efficient representation of local geometry for large scale object retrieval. In *CVPR*, 2009.

[93] P. Perona, R. Fergus, and F. F. Li. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. In *Workshop on Generative Model Based Vision*, 2004.

[94] James Philbin, Ondrej Chum, Michael Isard, Josef Sivic, and Andrew Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *CVPR*, 2007.

[95] James Philbin, Ondrej Chum, Michael Isard, Josef Sivic, and Andrew Zisserman. Lost in quantization: improving particular object retrieval in large scale image databases. In *CVPR*, 2008.

[96] T. Quack, V. Ferrari, B. Leibe, and L.V. Gool. Efficient mining of frequent and distinctive feature configurations. In *ICCV*, 2007.

[97] Andrew Rabinovich, Andrea Vedaldi, Carolina Galleguillos, Eric Wiewiora, and Serge Belongie. Objects in context. In *ICCV*, 2007.

[98] Pradeep Ravikumar and John Lafferty. Quadratic programming relaxations for metric labeling and markov random field map estimation. In *ICML*. ACM Press, 2006.

[99] B. C. Russell, W. T. Freeman, A. A. Efros, J. Sivic, and A. Zisserman. Using multiple segmentations to discover objects and their extent in image collections. In *CVPR*, 2006.

[100] M. S. Ryoo, Chia-Chih Chen, J. K. Aggarwal, and Amit Roy-Chowdhury. An overview of contest on semantic description of human activities (SDHA) 2010. In *ICPR Contests*, 2010.

[101] Michael S. Ryoo and Jake K. Aggarwal. Spatio-temporal relationship match: Video structure comparison for recognition of complex human activities. In *ICCV*, 2009.

[102] S. Savarese, A. Del Pozo, J. C. Niebles, and F. F. Li. Spatial-temporal correlatons for unsupervised action classification. In *WMVC*, 2008.

[103] C. Schuldt, I. Laptev, and B. Caputo. Recognizing human actions: a local SVM approach. In *ICPR*, 2004.

[104] Jamie Shotton, Matthew Johnson, and Roberto Cipolla. Semantic texton forests for image categorization and segmentation. In *CVPR*, 2008.

[105] Jamie Shotton, John M. Winn, Carsten Rother, and Antonio Criminisi. Textonboost for image understanding: Multi-class object recognition and segmentation by jointly modeling texture, layout, and context. *IJCV*, 81(1):2–23, 2009.

[106] Josef Sivic and Andrew Zisserman. Video google: a text retrieval approach to object matching in videos. In *ICCV*, 2003.

[107] J. Sun, X. Wu, S. C. Yan, L. F. Cheong, T. S. Chua, and J. T. Li. Hierarchical spatio-temporal context modeling for action recognition. In *CVPR*, 2009.

[108] Carlo Tomasi and Roberto Manduchi. Bilateral filtering for gray and color images. In *ICCV*, 1998.

[109] A. Torralba. Contextual priming for object detection. *IJCV*, 53(2):169–191, 2003.

[110] Antonio Torralba, Kevin P. Murphy, and William T. Freeman. Contextual models for object detection using boosted random fields. In *NIPS*, 2004.

[111] Lorenzo Torresani, Martin Szummer, and Andrew Fitzgibbon. Learning query-dependent prefilters for scalable image retrieval. In *CVPR*, 2009.

[112] T. Toyoda and O. Hasegawa. Random field model for integration of local information and global information. *PAMI*, 30(8):1483–1489, August 2008.

[113] I. Tsochantaridis, T. Hofmann, T. Joachims, and Y. Altun. Support vector machine learning for interdependent and structured output spaces. In *ICML*, 2004.

[114] Z. W. Tu. Auto-context and its application to high-level vision tasks. In *CVPR*, 2008.

[115] Manik Varma and Bodla Rakesh Babu. More generality in efficient multiple kernel learning. In *ICML*, volume 382, 2009.

[116] Andrea Vedaldi, Varun Gulshan, Manik Varma, and Andrew Zisserman. Multiple kernels for object detection. In *ICCV*, 2009.

[117] Daniel Waltisberg, Angela Yao, Juergen Gall, and Luc J. Van Gool. Variations of a hough-voting action recognition system. In *ICPR Contests*, Lecture Notes in Computer Science. Springer, 2010.

[118] G. Wang, Y. Zhang, and L. Fei-Fei. Using dependent regions for object categorization in a generative framework. In *CVPR*, 2006.

[119] H. Wang, M. M. Ullah, A. Klaser, I. Laptev, and C. Schmid. Evaluation of local spatio-temporal features for action recognition. In *BMVC*, 2009.

[120] Jinjun Wang, Jianchao Yang, Kai Yu, Fengjun Lv, Thomas S. Huang, and Yihong Gong. Locality-constrained linear coding for image classification. In *CVPR*, 2010.

[121] Ping Wang, Gregory D. Abowd, and James M. Rehg. Quasi-periodic event analysis for social game retrieval. In *ICCV*, 2009.

[122] X. G. Wang, X. X. Ma, and W. E. L. Grimson. Unsupervised activity perception in crowded and complicated scenes using hierarchical Bayesian models. *PAMI*, 31(3):539–555, 2009.

[123] Y. Weiss and W.T. Freeman. On the optimality of solutions of the max-product belief-propagation algorithm in arbitrary graphs. *Information Theory*, 47(2):736 –744, 2001.

[124] Y. Weiss and A. Levin. Learning to combine bottom-up and top-down segmentation. In *ECCV*, 2006.

[125] Yair Weiss, Antonio Torralba, and Rob Fergus. Spectral hashing. In *NIPS*, 2009.

[126] G. Willems, T. Tuytelaars, and L. J. Van Gool. An efficient dense and scale-invariant spatio-temporal interest point detector. In *ECCV*, 2008.

[127] S. F. Wong, T. K. Kim, and R. Cipolla. Learning motion categories using both semantic and structural information. In *CVPR*, 2007.

[128] Shandong Wu, Brian E. Moore, and Mubarak Shah. Chaotic invariants of

lagrangian particle trajectories for anomaly detection in crowded scenes. In *CVPR*, 2010.

[129] Zhong Wu, Qifa Ke, Michael Isard, and Jian Sun. Bundling features for large scale partial-duplicate web image search. In *CVPR*, 2009.

[130] J. C. Yang, K. Yu, Y. H. Gong, and T. S. Huang. Linear spatial pyramid matching using sparse coding for image classification. In *CVPR*, 2009.

[131] Yi Yang, Sam Hallman, Deva Ramanan, and Charless Fowlkes. Layered object detection for multi-class segmentation. In *CVPR*, 2010.

[132] Angela Yao, Jürgen Gall, and Luc J. Van Gool. A Hough transform-based voting framework for action recognition. In *CVPR*, 2010.

[133] Junsong Yuan, Ying Wu, and Ming Yang. Discovery of collocation patterns: from visual words to visual phrases. In *CVPR*, 2007.

[134] R. Zabih, O. Veksler, and Y. Y. Boykov. Fast approximate energy minimization via graph cuts. In *ICCV*, 1999.

[135] Matthew D. Zeiler, Graham W. Taylor, and Rob Fergus. Adaptive deconvolutional networks for mid and high level feature learning. In *ICCV*, 2011.

[136] Gloria Zen and Elisa Ricci. Earth mover's prototypes: A convex learning approach for discovering activity patterns in dynamic scenes. In *CVPR*, 2011.

[137] J. Zhang, S. Lazebnik, and C. Schmid. Local features and kernels for classification of texture and object categories: a comprehensive study. *International Journal of Computer Vision*, 73, 2007.

[138] Xiao Zhang, Zhiwei Li, Lei Zhang, Weiying Ma, and Heung-Yeung Shum. Efficient indexing for large scale visual search. In *ICCV*, 2009.

[139] Yimeng Zhang, Weina Ge, Ming-Ching Chang, and Xiaoming Liu. Group context learning for event recognition. In *WACV*, 2012.

[140] C. Lawrence Zitnick, Jie Sun, Richard Szeliski, and Simon Winder. Object instance recognition using triplets of feature symbols. *Tech. Report, Microsoft Research*, 2007.